ESD-TR-89-104

Technical Report
846

Efficient Detection of Small Moving Objects

P.L. Chu

21 July 1989

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

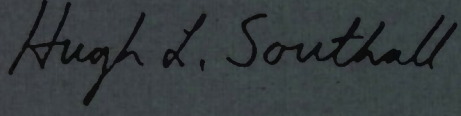
Prepared for the Department of the Air Force
under Electronic Systems Division Contract F19628-85-C-0002.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Air Force under Contract F19628-85-C-0002.

The ESD Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER



Hugh L. Southall, Lt. Col., USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

EFFICIENT DETECTION OF SMALL MOVING OBJECTS

P.L. CHU
Group 27

TECHNICAL REPORT 846

21 JULY 1989

LEXINGTON MASSACHUSETTS

ABSTRACT

A signal processing problem encountered with many sensor systems having a wide field-of-view is detection of small, unresolved objects moving in a straight line amid stationary clutter. The wide field-of-view combined with the need to accurately pinpoint object positions imply that these sensors must have hundreds of thousands of samples in their output. To process this amount of data in a timely fashion, computationally efficient algorithms are a necessity.

In this report, a computationally efficient set of algorithms is described for detecting satellites, meteorites, and other moving objects using data from an optical telescope charge-coupled device (CCD) focal plane in the MIT Lincoln Laboratory Demonstration Surveillance System (DSS). The trade-off of reduced detection sensitivity for lower computational cost in the algorithm is quantitatively discussed. Major techniques employed are:

1. Sample normalization by temporal mean and standard deviation to suppress clutter.
2. Maximum value projection to reduce the dimensionality of the data.
3. A two-stage matched filter detector which first nominates and then confirms signal candidates.
4. Two-dimensional binary velocity filtering.

The techniques should have practical application to other wide field-of-view sensors where moving object detection is important.

ACKNOWLEDGEMENTS

The author would like to thank Dr. A.E. Filip for many helpful discussions on all topics of this report. The two-dimensional binary velocity filter (Section 3.3) incorporates some suggestions from R.W. Burke. The sensitivity comparison of the MTI section of the algorithm to GEODSS MTI (Section 3.6.1) was done by G.S. Downs.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

1. INTRODUCTION

Detection of small moving objects in a time sequence of data is a problem encountered in surveillance applications such as the optical detection of satellites [1], infrared detection of dim moving objects [2,3,4], detection of objects using spectrograms from several acoustic channels [5,6], and other applications.

The combined requirements of wide field-of-view and accurate position estimates imply that these sensors must have many samples at their output, each sample representing the state of some small portion of the field-of-view at a given time. In many situations the apparent size of the object will be less than the resolution of the sensor, and the movement of the object will approximate a straight line during the observation. Therefore, detection algorithms must simply search for multidimensional line segments in the large volume of sensor data.

If computational cost is of no concern, the optimal approach, which gives the greatest probability of detection (p_d) for a fixed probability of false alarm (p_{fa}), is the multidimensional matched filter [7]. This approach involves summing up sample values in all possible multidimensional line segment paths (i.e., velocity vectors) and subsequently comparing these summations to detection thresholds. Unfortunately, if the volume of data is huge, then the number of velocity vectors to consider is also huge, resulting in prohibitive computational costs.

Some authors have attempted to perform the full-dimensional path search in computationally efficient ways. Blostein and Huang [8] organize the paths in a tree structure and perform a Wald-type sequential probability ratio test procedure to eliminate having to search through all branches of the tree. Barniv [3] uses a dynamic programming approach to search through the paths. Porat et al. [2] use a full-dimensional bank of filters in the spectral domain (a full-dimensional FFT is required) to do the various path summations. Unfortunately, all of these methods are much too computationally expensive for the amount of data and possible object velocities in the MIT Lincoln Laboratory DSS.

Thus, there is a strong motivation to use suboptimal detection algorithms which will have less sensitivity than the optimal algorithm but will be much easier to implement. The philosophy of algorithm development has been to first develop signal and noise models which closely approximate reality yet are simple enough to be used in analysis. Next, these models are used to derive optimal detection procedures using classical detection theory. If the procedures are computationally prohibitive, then suboptimal procedures are created. At all times the trade-off between computational cost and detection sensitivity is kept in mind. Finally, the algorithm is tested on real data and modified appropriately until satisfactory performance is achieved.

The particular sensor for which the algorithm was developed is a 420 x 420-pixel frame-transfer optical CCD produced at MIT Lincoln Laboratory. Multiple CCDs mounted adjacent to one another are placed in the focal plane of a telescope. Figure 1-1 is a block diagram of the entire system.

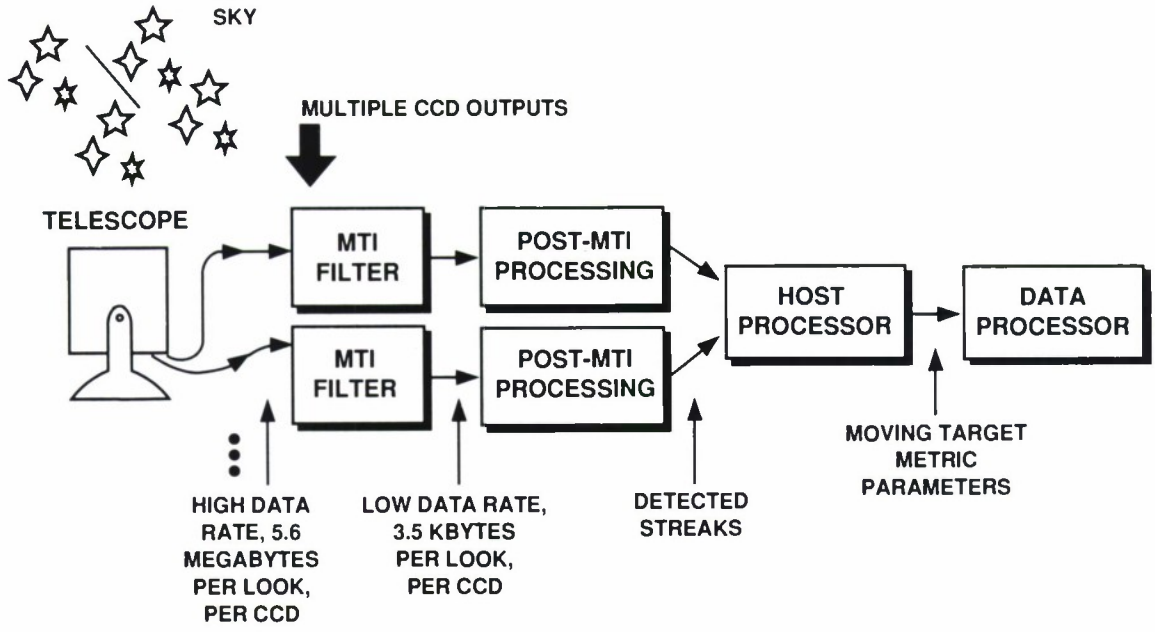


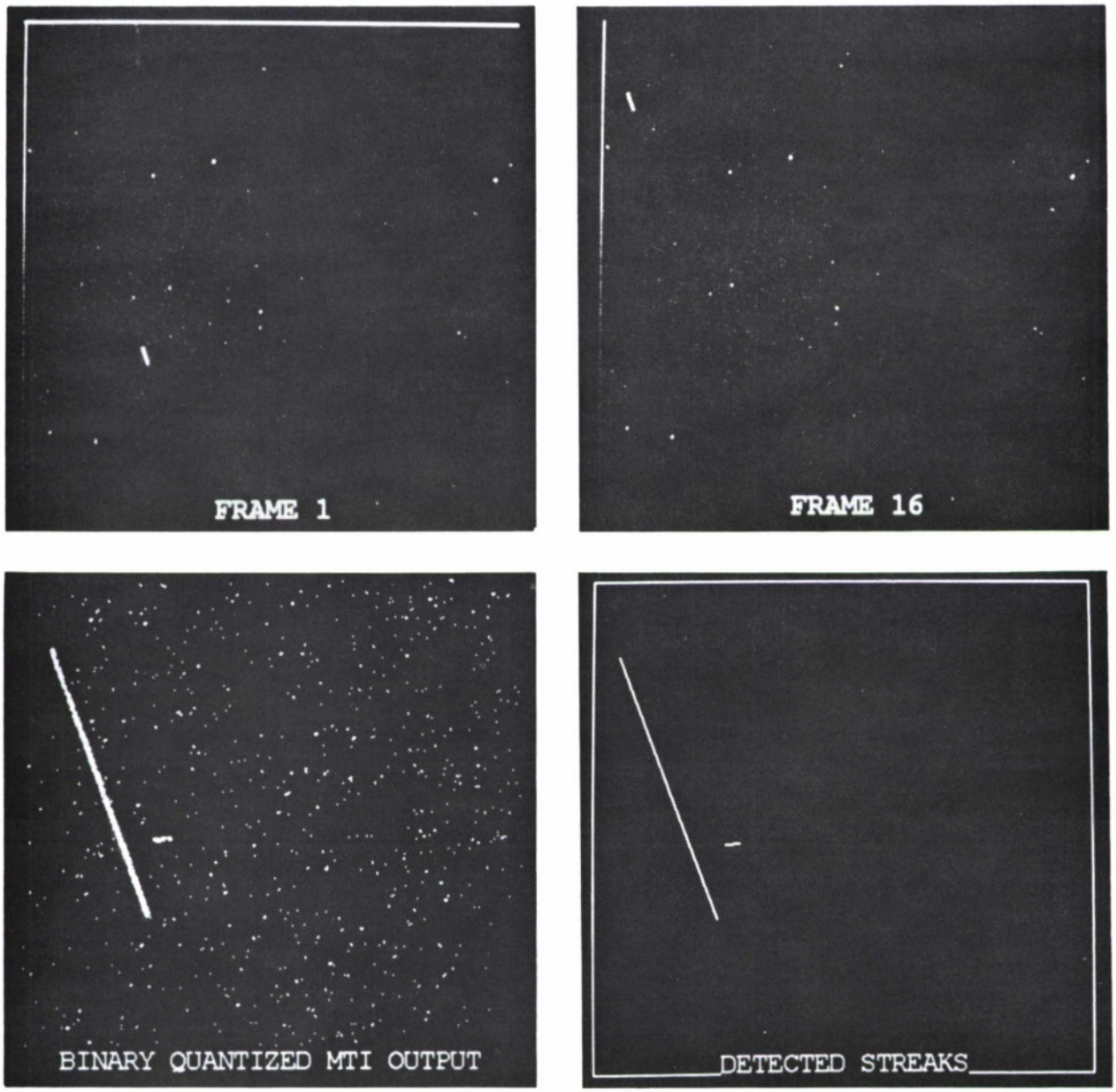
Figure 1-1. Block diagram of the optical detection system.

The telescope generates an image of the sky on a CCD focal plane. The equatorial telescope mount can be used either in sidereal track mode where stars appear stationary or target track mode where stars appear as streaks. The mount step- and settle-time is about 2 s. It is desirable for the detection processing to be completed within this time interval.

The CCDs are exposed for intervals of time ranging from 0.2 to 12.8 s, and the pixel charges are subsequently read out at a 1-MHz rate. The output from each CCD array goes to its own detection processor. The voltages are quantized to 12 bits by an A/D converter and are specified by the position of the CCD cell on the focal plane and the frame number corresponding to the time when the voltage was sampled. The amount of data produced per look is huge, being as much as 16 frames, each frame consisting of 176,400 pixels for each CCD array. In Figure 1-2 are shown the first and last frames from a 16-frame set of 420 x 420-pixel CCD data.

The detection processing is split into two sections. The first section, the Moving Target Indicator (MTI) filter, performs several simple operations on each sample as it arrives at the input to the processor. Because these operations must be performed relatively quickly, the MTI filter is implemented using special-purpose hardware. In the hardware implementation we have built, all MTI operations are completed about 1 s after the completion of data input. The MTI filter eliminates objects moving at a specified velocity (usually zero) while projecting the multiframe data along the time axis onto a single frame, which is binary quantized. The volume of binary quantized data is hundreds of times less than the original input data.

The much reduced data set at the output of the MTI filter is sent to the second section of the



114253-2

Figure 1-2. First and last frames of a 16-frame set of 420 x 420-pixel CCD data, binary quantized MTI output and detected streaks.

detection algorithm, the post-MTI processor, and implemented with a Motorola 68020 microprocessor. The relatively complex post-MTI algorithm examines the binary projection of the data and performs binary velocity filtering and full-precision estimation of streaks corresponding to moving objects. Real-time implementation (approximately 1 s allowed for post-MTI processing) can be achieved using a single 68020 processor per CCD chip. The parameters of detected moving objects are fed to a host processor, which examines the detected streaks of all the CCD arrays and looks for streaks which cross from one CCD array to another CCD array. The host processor sends its list of detected streaks to the data processor, which performs the various mission functions. These functions include maintenance of a catalog of known objects, measurement of changes in the orbits of these objects, detection and orbit determination of new objects.

Although the detection algorithms are presented in the context of DSS, they are general enough so that, with appropriate modifications, they may be of use in any scenario where small moving targets are being detected in a time sequence of data. Indeed, recent experiments have been successfully performed in which the system is operated in object track mode (objects appear as points and stars are streaked). Only minor changes to the software were needed to accommodate the new system operating mode, which is discussed in section 2.8.

As the detection algorithm is split into two sections, this report is split into two parts, the first dealing with the MTI processor and the second dealing with the post-MTI processor.

2. THE MTI FILTER

2.1 INTRODUCTION TO THE MTI FILTER

The MTI filter has for its input a number of frames of CCD data. These frames form a time sequence consisting of moving objects, clutter (stars, galaxies, etc.), Gaussian distributed noise (from the amplifier circuit), and Poisson distributed noise, all added together.

A block diagram of the algorithm is shown in Figure 2-1. First, the MTI filter forms a single frame of data consisting of the maximum value in each pixel found among 2 to 16 frames of input data. The maximum number of frames is constrained by the hardware, not the algorithm. The middle channel in Figure 2-1 is the pixel average computed across the frames, while the bottom channel derives an estimate of the pixel standard deviation. Combining the three channels as shown yield a single frame where streaks corresponding to moving objects are retained, while nonmoving objects are absent or greatly attenuated. The final output of the MTI filter is a binary quantized image of the projection upon which true matched filter detection is performed in the post-MTI processor. An exhaustive matched filter velocity search is easily implemented because operations occur only on the "1" pixels. The only candidate streak paths considered are those with "1" pixels for their endpoints. If the number of "1" pixels is small the number of these paths is also small.

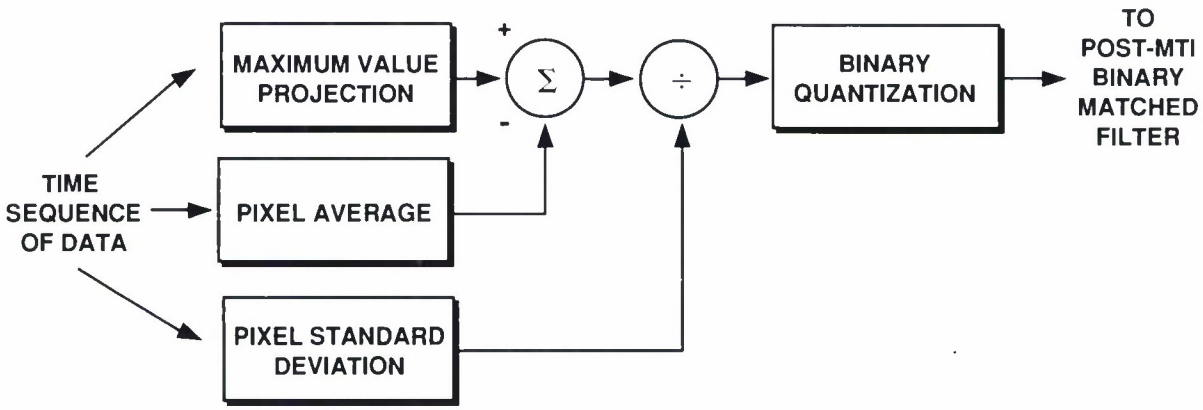


Figure 2-1. Block diagram of MTI filter.

In section 2.2 the signal and noise models will be discussed, supplying a basis for the algorithmic issues. The signal and noise models are used to derive the optimal full-dimensional matched filter detector in section 2.3. The justification for maximum value projection is discussed in section 2.4. In section 2.5 the normalization of the maximum value projection by pixel mean and standard deviation is explained. Section 2.6 discusses the method of pairwise binary quantization, which supplies a 2-dB increase in detection sensitivity over single sample binary quantization. In section 2.7 the steps of the MTI algorithm are briefly summarized without explanation to present a complete but concise picture. In section 2.8 we illustrate how the MTI algorithm can be modified to eliminate

background clutter moving at any *known* velocity, accommodating both sidereal and target track modes of the telescope mount. The sensitivity of the MTI algorithm is compared to the GEODSS MTI algorithm in section 3.6.1.

2.2 SIGNAL AND NOISE MODELS

During a frame exposure, photons hit a CCD pixel cell and generate electrons. The electrons are shifted out during the read cycle to an amplifier whose output is a voltage proportional to the number of electrons at its input. Using a 12-bit A/D converter, the output voltages of the amplifier are converted to digital numbers which are processed by the detection algorithms.

Denote the number of electrons produced by a pixel at position x and frame t as $r(x,t)$. While position on the focal plane is conventionally specified in two dimensions, for ease of notation the position will only be specified in the single-dimensional variable x which corresponds to the linear address of a pixel in a one-dimensional array assuming the image is read out in a raster scan manner. Then the pixel electron count is the sum of four terms

$$r(x,t) = c(x,t) + d(x,t) + s(x,t) + n(x,t) \tag{2.1}$$

where $c(x,t)$ is the circuit induced Gaussian noise, $d(x,t)$ is the dark current and background light induced Poisson noise, $n(x,t)$ is noise from nonmoving objects, and $s(x,t)$ is the signal to be detected, i.e., the moving object. The large majority of the pixels will not have the nonmoving object component, since these objects (stars, galaxies, and nebula) are relatively rare events. Without this component the probability density of the combined noise is the convolution of Poisson and Gaussian densities, which is rather cumbersome for analysis. In section 2.4 to simplify analysis, the shape of the Poisson-Gaussian convolution is approximated as pure Gaussian, and the Poisson signal component is approximated as an additive dc offset.

The detection statistics are functions of the input signal-to-noise ratio (SNR) which is defined in units of decibels as

$$20\log_{10}\left[\frac{(\text{mean of signal+noise}) - \text{mean of noise}}{\text{standard deviation of noise}}\right] \tag{2.2}$$

for a single pixel of the moving object at the input to the MTI filter. The amplitude SNR is simply the bracketed quantity in (2.2).

2.2.1 Circuit Noise

The Gaussian distributed noise is independent, identically distributed (i.i.d.) from pixel to pixel and has probability density

$$p(c(x,t)) = \frac{1}{\sqrt{2\pi}\sigma} e^{-c(x,t)^2/\sigma^2} \tag{2.3}$$

where $c(x,t)$ denotes the Gaussian portion of the pixel charge at location x on the focal plane for frame t. The standard deviation of the Gaussian noise is σ in (2.3) and is known a priori by measurement of the CCD characteristics. The primary source of the noise is the circuitry which converts the CCD electron charges to voltages high enough to drive the A/D converter. The standard deviation is usually specified as if it were being added at one point directly to the electrons of the CCD charge packet before amplification. For Lincoln Laboratory CCDs, a standard deviation as low as 5 rms electrons has been measured for the circuit noise.

2.2.2 Poisson Distributed Noise

The Poisson distributed noise is due to background light and the inherent dark current of the CCD. While it is i.i.d. over time at a given location x, the Poisson parameter λ varies with x in an unpredictable manner due to nonuniformities in the CCD cells and background illumination. The mean and variance of the Poisson distribution is equal to λ. The probability distribution of the Poisson portion of the pixel electron count is

$$p(d(x,t)) = \frac{e^{-\lambda(x)}\lambda(x)^{d(x,t)}}{d(x,t)!} \tag{2.4}$$

where $\lambda(x)$ denotes that the λ parameter is a function of position, and $d(x,t)$ is a nonnegative integer corresponding to the number of electrons in the pixel charge packet due to the Poisson distribution. Assuming that no other processes are present in the pixel charges, the maximum likelihood estimator of $\lambda(x)$ is simply the average of the pixel electron count at location x over time t as in

$$\hat{\lambda}(x) = \frac{1}{N}\sum_{t=1}^{N} r(x,t) \tag{2.5}$$

where N is the number of frames of data. In reality since other processes will be present in unknown proportions, (2.5) may not always be the best estimator, but because of its computational simplicity it is employed in the algorithm.

2.2.3 Nonmoving Objects

Unfortunately, there is an erratic portion of the pixel charge $n(x,t)$ due to nonmoving objects like stars, galaxies, etc. If there were no atmospheric turbulence or mount jitter, these objects could then be viewed as nonuniformities in the background illumination, and the Poisson model of (2.4) would be valid. However, the presence of jitter randomly perturbs the position of the object on the focal plane, producing variations in illumination which are difficult to accurately model. For the most part the pixel electron counts have probability densities with large tails (relative to their standard deviation) and wide variability with position. For the idealized case where the nonmoving

object is a point source exactly centered in a pixel, and the jitter and point spread function of the optics are both Gaussian distributed, Fried [9] has shown that the illumination on the pixel will have a log-normal distribution.

The following example illustrates why nonmoving objects produce illuminations with widely varying statistics. Assume the nonmoving object is a uniformly lit disk of several pixels diameter, and that the jitter is less than a pixel in extent. Then, pixels in the center of the disk will be lit with constant intensity and the resulting electron count will have Poisson distribution, while pixels right at the disk edge will have varying amounts of light from the disk dependent on the jitter movement at the particular time during which the frame is exposed and will have unpredictable distributions. Therefore, the illumination distributions due to the disk will be a strong function of proximity to the disk's edge.

2.2.4 Moving Objects

Detection and metric parameter estimation of moving objects is the end goal of the algorithm. A moving object is classified as an object with velocity greater than or equal to a pixel per frame in any arbitrary direction. In general, a moving object may vary dramatically in brightness as a function of time, a case in point being a rotating satellite with large solar cell panels. It is difficult to exploit the variation in brightness without a priori information. Therefore, to simplify the algorithm the object is assumed to be of constant brightness, except in the final stages of parameter estimation (after detection) where amplitude information is easily incorporated.

Suppose that the object passes through position X in frame T, then it is assumed that $r(X,t)$ for $t \neq T$ has no contribution from the object. In practice if X is the position of the first occurrence of the object in frame T, then $r(X,T-1)$ may have significant contribution from the object; while if X is the position of the last occurrence of the object in frame T, then $r(X,T+1)$ may have significant contribution from the object. However, to reduce algorithm computations the simplified model is used in algorithm derivations.

Define $\lambda_s(x)$ as the spatial distribution of the intensity of the object at $t = 0$. For most objects $\lambda_s(x)$ will appear to be a streak with length equal to the speed of the object in pixels per frame. The spatial distribution of the intensity at some other value of t is simply a spatially shifted version of the spatial distribution, $\lambda_s(x - v_x t - nrow \cdot v_y t)$, where v_x is horizontal velocity of the object in pixels per frame, v_y is the vertical velocity of the object in pixels per frame, and $nrow$ is the number of pixels in a row on the CCD. The two spatial dimensions of the CCD are represented by the single-dimensional variable x, which specifies spatial position in a raster scan fashion. We only allow these velocities to be integers for our simplified model, although, in general, they will be continuous quantities. Unfortunately, allowing continuous velocities result in the derivation of detectors which would require spatial interpolation, which is a computationally expensive operation. The $s(x,t)$ contribution from the object will be a spatially dependent Poisson distributed random quantity of the form

$$p(s(x,t)) = \frac{e^{-\lambda_s(x-v_xt-nrow\cdot v_yt)}\lambda_s(x-v_xt-nrow\cdot v_yt)^{s(x,t)}}{s(x,t)!}. \tag{2.6}$$

2.3 OPTIMAL FULL-DIMENSIONAL DETECTOR

Consider the case where the dominant source of noise is dark current and uniform background light, as discussed in section 2.2. The nonmoving objects of section 2.3 are ignored since they occur infrequently, and an heuristic measure of standard deviation can handle their occurrence as is discussed in section 2.5.

The log likelihood ratio detector [10] achieves the highest probability of detection for a fixed probability of false alarm and is of the form

$$\log\left[\frac{p(\ r(x,t) \text{ for all } x,t \mid H_{sig})}{p(\ r(x,t) \text{ for all } x,t \mid H_{noise})}\right] \overset{?}{>} T. \tag{2.7}$$

Under the signal hypothesis the samples will have the sum of the signal and noise contributions

$$\mathrm{p(r(x,t) \mid H_{sig}) = \frac{(\lambda(x)+\lambda_s(x-v_xt-nrow\cdot v_yt))^{r(x,t)}\, e^{-\lambda(x)-\lambda_s(x-v_xt-nrow\cdot v_yt)}}{r(x,t)!}} \tag{2.8}$$

and for the noise hypothesis, the samples will contain the noise contribution alone

$$p(r(x,t) \mid H_{noise}) = \frac{\lambda(x)^{r(x,t)}e^{-\lambda(x)}}{r(x,t)!}. \tag{2.9}$$

Combining (2.7), (2.8), and (2.9), the optimal detector is

$$\sum_x \left(\sum_{t=1}^{N} r(x+v_xt+nrow\cdot v_yt,t)\log(1+\frac{\lambda_s(x)}{\lambda(x+v_xt+nrow\cdot v_yt)})\right) \overset{?}{>} T_1 \tag{2.10}$$

where constant terms have been absorbed in the threshold T_1. For the weak signal case, where $\lambda_s(x)$ is small compared with $\lambda(x+v_xt+nrow\cdot v_yt)$, (2.10) may be approximated as

$$\sum_x \left(\lambda_s(x)\sum_{t=1}^{N}\frac{r(x+v_xt+nrow\cdot v_yt,t)}{\lambda(x+v_xt+nrow\cdot v_yt)}\right) \overset{?}{>} T_2 \tag{2.11}$$

where constants have been absorbed in T_2. The interpretation of (2.11) is that the optimal detection procedure is to divide each sample by the variance of the noise, shift the data at the inverse of the velocity of the object to be detected, sum the shifted frames together, and subsequently perform a spatial correlation between the accumulated frame and the signal intensity. A computational disadvantage of this approach is that for a given false alarm rate, the threshold T_2 in (2.11) is a function of the sum of the sample variances in the path, and, in general, must be recalculated for each path.

If the background is assumed uniform, which is a good approximation for many cases of signal detection, then the weighting term in (2.11) may be deleted, yielding

$$\sum_x \left(\lambda_s(x) \sum_{t=1}^{N} r(x + v_x t + nrow \cdot v_y t, t) \right) \stackrel{?}{>} T_3. \tag{2.12}$$

The threshold T_3 for a given false alarm rate is only a function of the total number of samples in the path, so it can be stored as a look-up table. If the noise is approximated as additive, white Gaussian noise and the signal as a dc offset, then a detector of the form of (2.12) also results. Covell [11] has explored optimal detection for mixed Poisson and Gaussian noise and found that the difference in sensitivity between (2.12) and a detector derived assuming mixed statistics to be negligible, less than a few tenths of a decibel.

2.4 PROJECTION

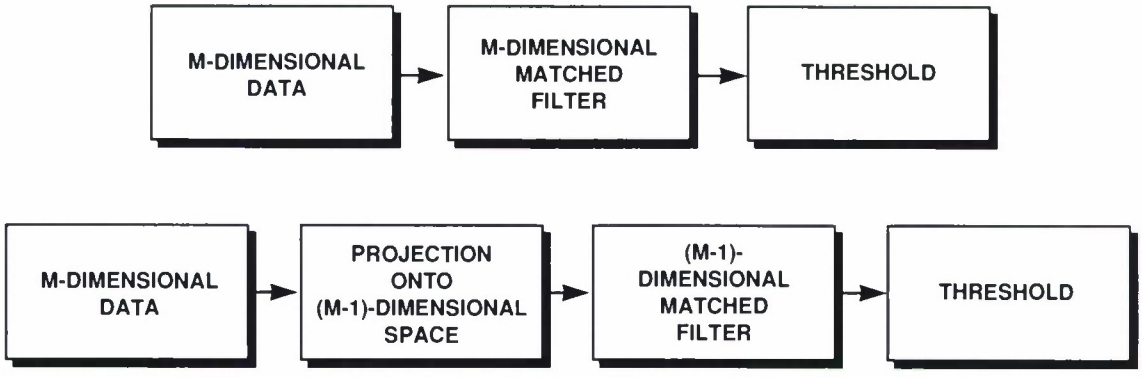
2.4.1 Optimum, Maximum Value, and Summation Projection

By projecting multidimensional data onto a lower-dimensional space, the number of samples to process and possible signal patterns are both greatly reduced. Matched filtering on the lower-dimensional space may now become computationally feasible. The cost of the reduced computation is a reduction in detection performance compared to multidimensional matched filtering. The two schemes are shown schematically in Figure 2-2.

The CCD noise model incorporates both Poisson and Gaussian noise sources. It is well known that the Poisson distribution converges to a Gaussian distribution for large counts [12]. We approximate the combined addition of Poisson and Gaussian noise as purely Gaussian noise, thereby simplifying analysis. Since the optimal projection scheme for additive, white Gaussian noise has been derived in [17], it is merely defined in this paper.

Using the same notation for the received data as in (2.1), denote the data as $r(x,t), x = 1, ..., L, t = 1, ..., N$. For each x, the set of samples, $\{r(x,t), t = 1, ..., N\}$, will be projected onto a single sample $z(x)$.

Those samples containing only noise are i.i.d. with probability density $p_{noise}(r(x,t))$, and those samples containing the signal and noise are i.i.d. with probability density, $p_{sig}(r(x,t))$. The set $\{r(x,t), t = 1, ..., N\}$ may include the signal in one sample, such as a modeling assumption met in



117435-1

Figure 2-2. Full-dimensional matched filter detection and projection followed by matched filtering detection block diagrams.

applications where a small moving object is being detected. No other restrictions are required of the probability densities. Thus, the noise may be either multiplicative or additive and have any arbitrary distribution shape, e.g., Gaussian, Poisson, Rayleigh, or Laplacian. However, the reader is forewarned that a closed-form solution of the optimum projection scheme may not exist for many types of noise.

The signal hypothesis is that a constant-intensity signal is present in the M sets $\{r(x,-)\}, x = l_1, l_2, ..., l_M$. The optimal projection operator is

$$z(x) = \log\left[\sum_{t=1}^{N} \frac{p_{sig}(r(x,t))}{p_{noise}(r(x,t))}\right]. \tag{2.13}$$

The projected samples are fed into a matched filter whose output is thresholded to decide whether or not a signal is present.

$$\sum_{x=l_1,...,l_M} z(x) \overset{?}{>} T. \tag{2.14}$$

For additive, white Gaussian noise, the noise probability density is

$$p_{noise}(r(x,t)) = \frac{1}{\sqrt{2\pi}} e^{-r(x,t)^2/2} \tag{2.15}$$

and for the samples containing signal

$$p_{sig}(r(x,t)) = \frac{1}{\sqrt{2\pi}} e^{-(r(x,t)-S)^2/2} \tag{2.16}$$

where S is the unprojected SNR of the signal. If (2.15) and (2.16) are substituted into (2.13), the projection function becomes

$$z(x) = \log \left[\sum_{t=1}^{N} e^{S \cdot r(x,t)} \right] \tag{2.17}$$

ignoring a constant offset which has been absorbed in the threshold of the matched filter of (2.14).

Optimal projection is cumbersome computationally because of the need to estimate the SNR of the object and evaluate transcendental functions (or represent these functions as look-up tables). When S in (2.17) is large (implying a large SNR), the particular $r(x,t)$ which is largest in value will dominate the summation in (2.17), and the log operation will simply produce a value for the projected sample close to that of the largest $r(x,t)$ so that

$$z(x) \approx \max[r(x,t), t = 1, ..., N] \tag{2.18}$$

giving rise to maximum value projection where the maximum value of a set of samples is the projected sample.

Previous work on maximum value projection may be found as early as 1968 [13]. More recent works [6,14,15,16,5] have called this method of projection "ORing" and have studied its application toward reducing the amount of underwater acoustic data for visual observation by a human. Struzinski [6,14,15] has studied the performance characteristics of "ORing" using first and second moments and a Gaussian approximation to non-Gaussian distributions. This procedure does not supply enough accuracy for our application. Nuttall [16] explicitly (without approximations) calculated performance characteristics. Using Monte Carlo techniques, Bottomley [5] studied performance characteristics of maximum value projection when the noise was correlated.

When S is small (relative to the noise standard deviation) in (2.17), the argument of the exponent $S \cdot r(x,t)$ will be small. Equation (2.17) may then be approximated as

$$z(x) \approx \sum_{t=1}^{N} r(x,t). \tag{2.19}$$

For additive, white Gaussian noise, the noise variance of the projected sample is N times higher than an unprojected sample, while the signal dc component is the same for both cases. Therefore, the SNR in the projected sample is degraded by $20 \log_{10} \sqrt{N}$ dB compared to the unprojected sample.

2.4.2 Evaluation of the Projection Algorithms

To decide whether or not a projection scheme followed by matched filtering is useful, its loss in performance compared to full-dimensional matched filtering must be known. For additive,

white Gaussian noise, suppose it is found that a certain unprojected input SNR is required to produce given probabilities of detection and false alarm using a full-dimensional matched filter. As described in section 2.3, this filter simply sums together the pixels in the full-dimensional space along an hypothesized velocity vector and checks to see if the sum exceeds a threshold. Then the loss in performance of a projection scheme followed by lower-dimensional matched filtering can be measured by finding the increase in the input SNR required for the projection scheme to produce equal probabilities of detection and false alarm.

For the full-dimensional matched filter it is assumed that t, the location of the signal containing sample in the set being projected, $\{r(x,t), t = 1, ..., N\}$, is known, whereas in deriving the optimal projection scheme, t was hypothesized as being unknown. Using the additive, white Gaussian noise and constant signal models of (2.15) and (2.16), if the signal is present in M sets of $\{r(x,-)\}$, the SNR in decibels at the output of the full-dimensional matched filter is $20\log_{10}(\sqrt{M}\ \mathrm{SNR}_{\mathrm{unproj}})$ where $\mathrm{SNR}_{\mathrm{unproj}}$ is the unprojected input SNR in amplitude. If summation projection is implemented on the M sets with subsequent one-dimensional matched filtering, the one-dimensional matched filter output SNR will be $20\log_{10}(\sqrt{M}\ \mathrm{SNR}_{\mathrm{unproj}}) - 20\log_{10}\sqrt{N}$, where N is the number of samples in the set being projected. Thus, for equal probabilities of detection and false alarm, full-dimensional matched filter detection requires an input SNR $20\log_{10}\sqrt{N}$ dB lower than that required for summation projection followed by lower-dimensional matched filter detection.

The projection loss using the optimum scheme of (2.17) was found using Monte Carlo simulations. To evaluate projection loss using maximum value projection, the computer was used to convolve and integrate the associated probability densities.

In Figure 2-3 are plotted the losses in additive, white Gaussian noise of the three projection schemes of this paper as a function of the unprojected SNR for projections of 5 frames (t has 5 values in $r(x,t)$). Shown in Figures 2-4 and 2-5 are the losses for 10 and 20 frames, respectively. Fortunately, the loss is a very weak function of probability of detection or false alarm but a very strong function of unprojected SNR. For optimum projection, the loss varies only 0.2 dB or so for probabilities of detection from 0.85 to 0.95 and probabilities of false alarm from 10^{-2} to 10^{-4} (we were limited to evaluating a minimum p_{fa} of 10^{-4} by a computational limit of about 10^6 Monte Carlo runs). For maximum value projection, the curves vary a few tenths of a decibel for probabilities of detection from 0.85 to 0.95 and probabilities of false alarm from 10^{-2} to 10^{-10}.

Figures 2-3, 2-4, and 2-5 indicate that at high SNRs, the loss in performance of the optimum and maximum value projection schemes becomes small, while at low SNRs the loss is quite substantial. Because the loss of even the optimal projection scheme is too severe at low signal strengths, projection will only be useful at moderate signal strengths, i.e., greater than 6 dB or so. For this situation maximum value will deliver performance nearly identical to optimal projection but with significantly less computational complexity.

Note that at SNRs higher than about 6 dB, maximum value is virtually equivalent to optimum projection in performance, while at SNRs around -6 dB, summation projection is roughly equivalent to optimum projection in performance. A graphical method of seeing these equivalences is to plot the decision boundaries for the three projection methods, as is done in Figure 2-6. For this example

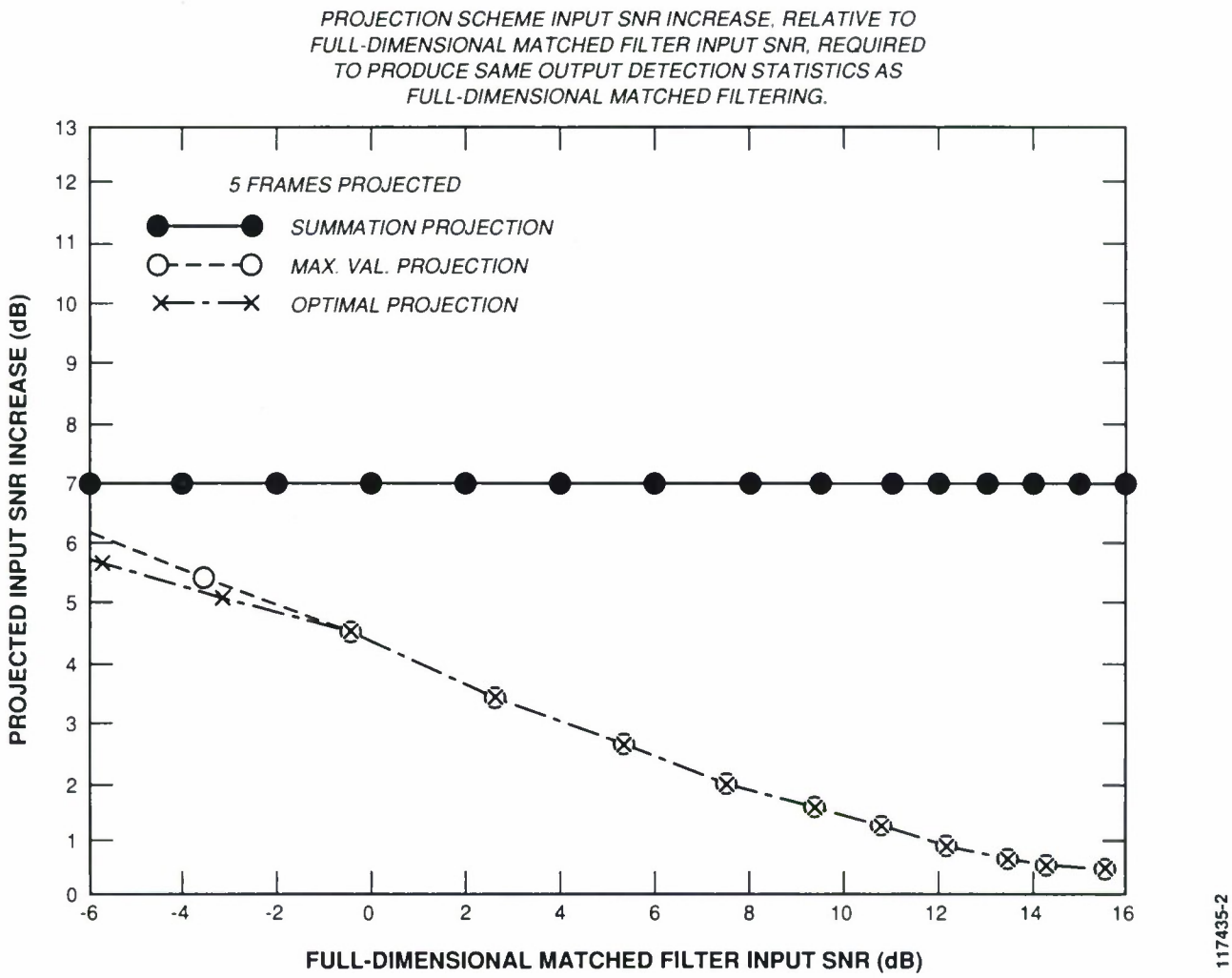


Figure 2-3. Projection scheme input SNR increase, relative to full-dimensional matched filter input SNR, required to produce equivalent output detection statistics as full-dimensional matched filtering (5 frames).

there are only two variables, x and y, with additive, white Gaussian noise and the signal may be present in only one of these variables. The optimal projection approach is to perform the following test for signal detection

$$\log\left(e^{Sx} + e^{Sy}\right) \overset{?}{>} T_1 \tag{2.20}$$

where S is the SNR. In Figure 2-6 are plotted three decision boundary curves for SNRs of -20, -6, and 6 dB (0.1, 0.5, 2 in amplitude). Also in Figure 2-6 are the decision boundary curves for

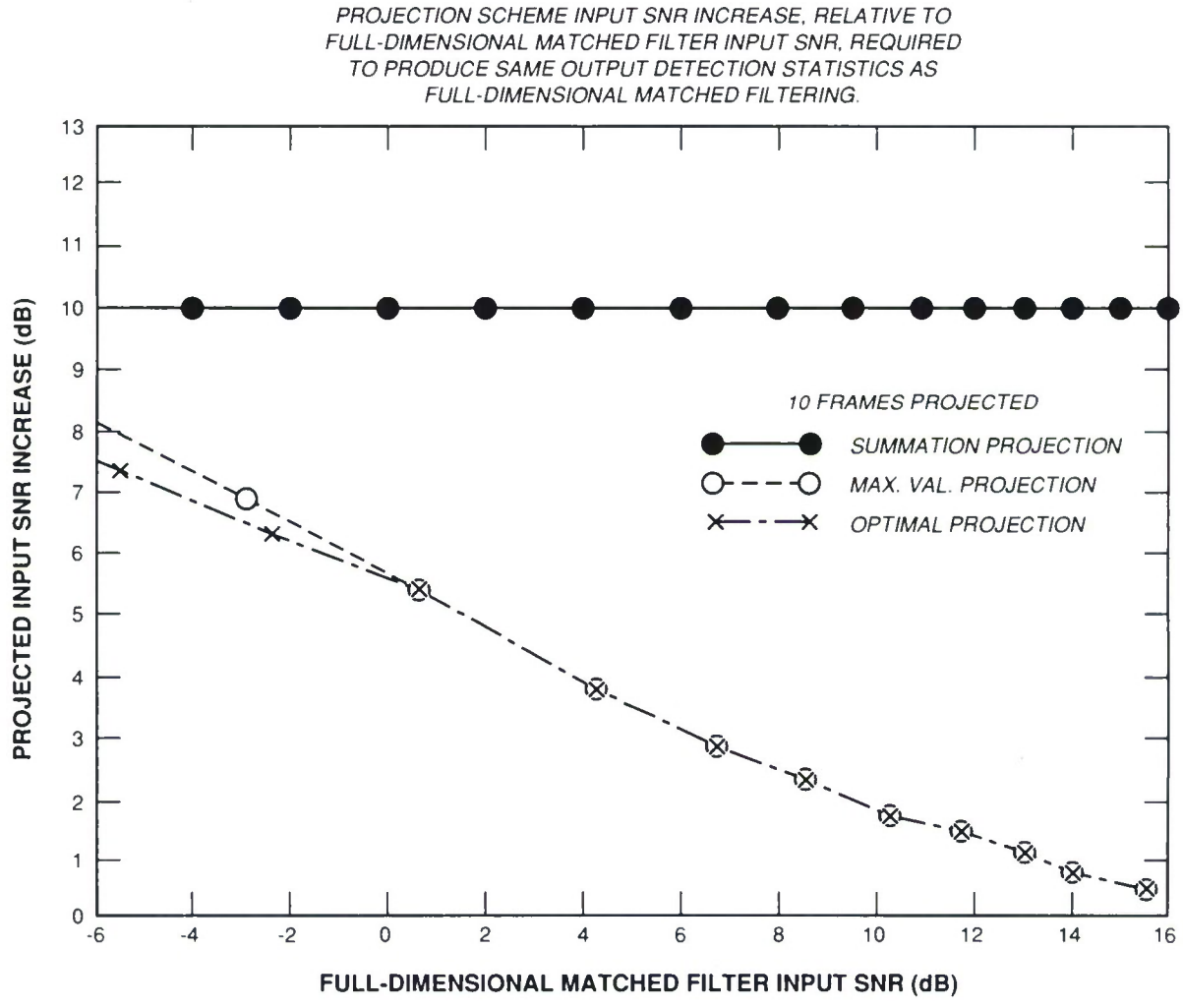


Figure 2-4. Projection scheme input SNR increase, relative to full-dimensional matched filter input SNR, required to produce equivalent output detection statistics as full-dimensional matched filtering (10 frames).

summation projection

$$x + y \overset{?}{>} T_2 \tag{2.21}$$

and maximum value projection

$$\max[x, y] \overset{?}{>} T_3. \tag{2.22}$$

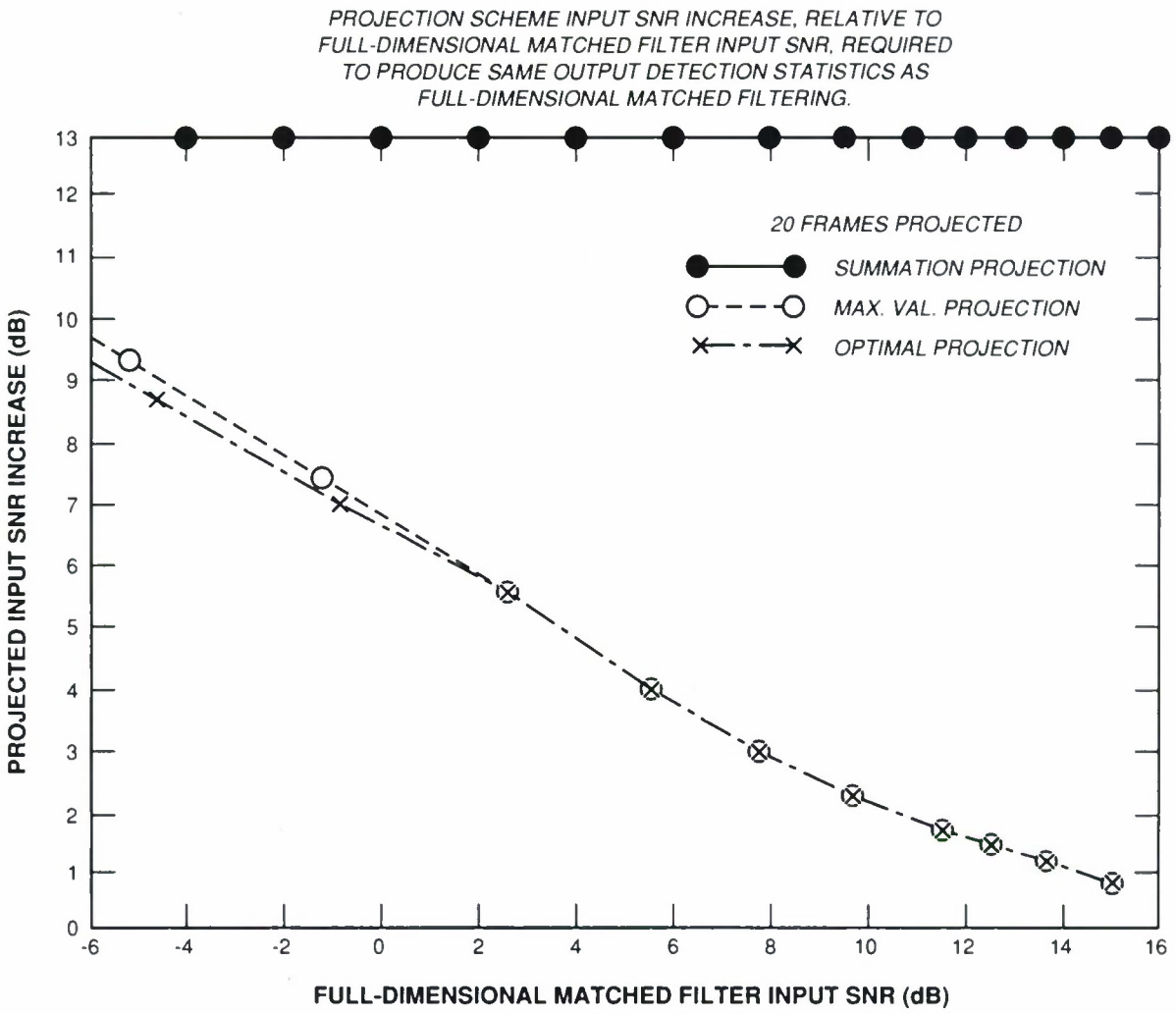
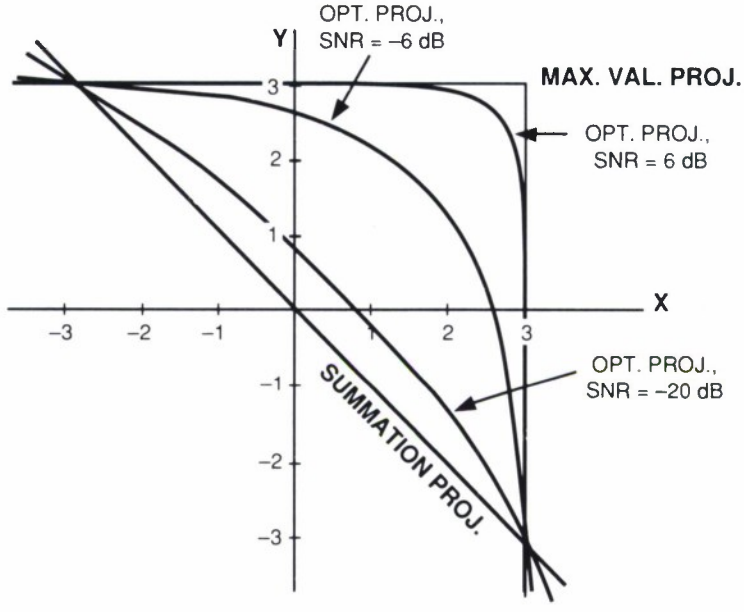


Figure 2-5. Projection scheme input SNR increase, relative to full-dimensional matched filter input SNR, required to produce equivalent output detection statistics as full-dimensional matched filtering (20 frames).

The thresholds T_1, T_2, and T_3 have been adjusted to make the curves coincide at (3,-3) and (-3,3) in the plot. Clearly for the SNR of 6 dB, the decision boundary curve for optimal projection is quite close to the curve for maximum value projection; while for the much lower SNR of -20 dB, the curve is quite close to the summation projection curve.

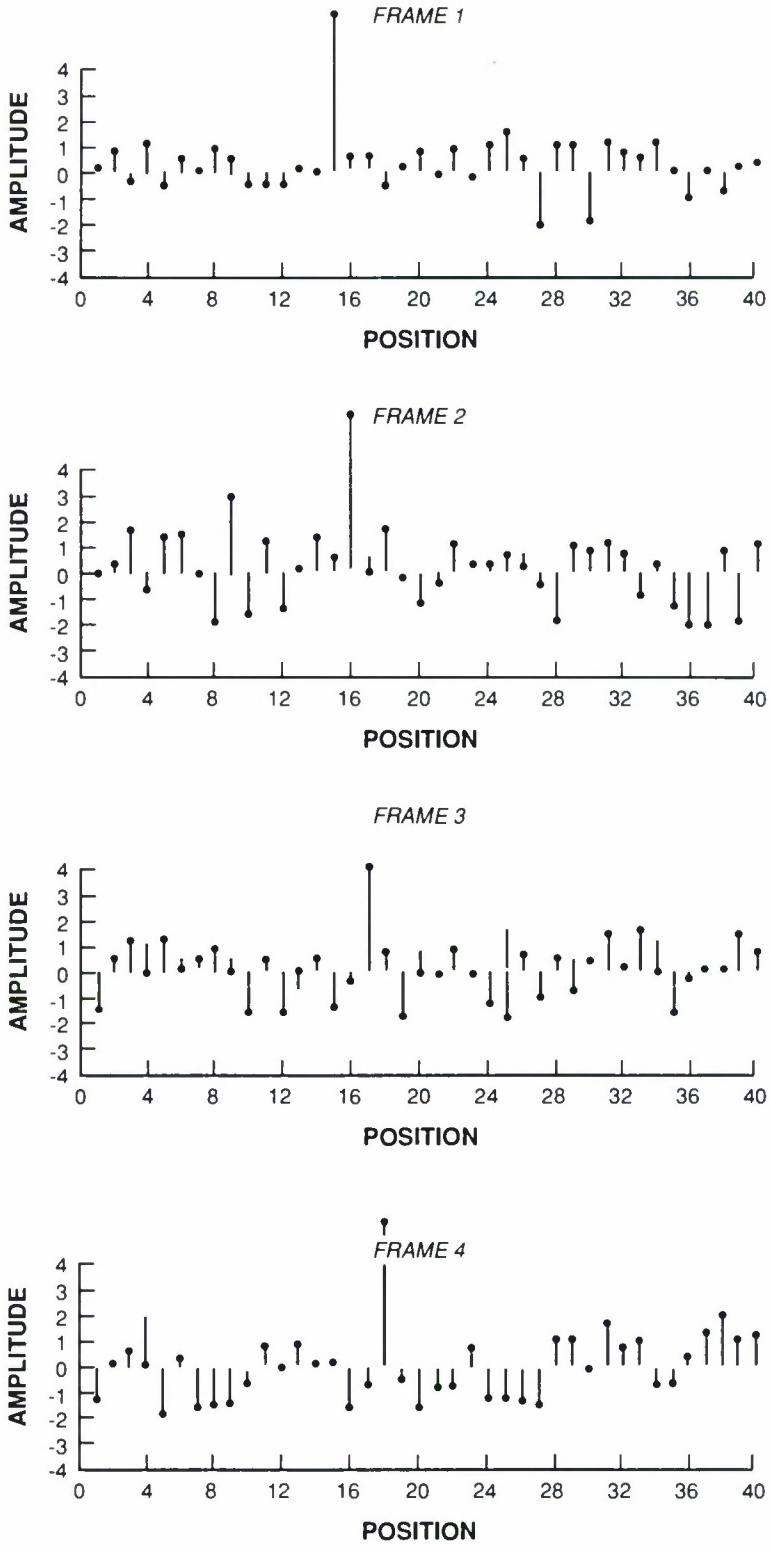
Another method of graphically seeing the differences between the projection algorithms is to the plot the projections for a simple data set. In Figure 2-7 are the summation projection, maximum value projection, and optimal projection of 9 frames of additive, normal Gaussian noise (standard



117435-15

Figure 2-6. Boundary decision curves for maximum value projection, optimal projection, and summation projection.

deviation of one). The signal has a constant SNR of 14 dB (5 in amplitude), occupies one sample in each of the frames, and moves one sample per frame.



117435-5

Figure 2-7. Summation, maximum value, and optimal projection of 9 frames, additive, white, normal Gaussian noise, signal occupies one sample with SNR 14 dB (5 in amplitude) moving at one sample per frame.

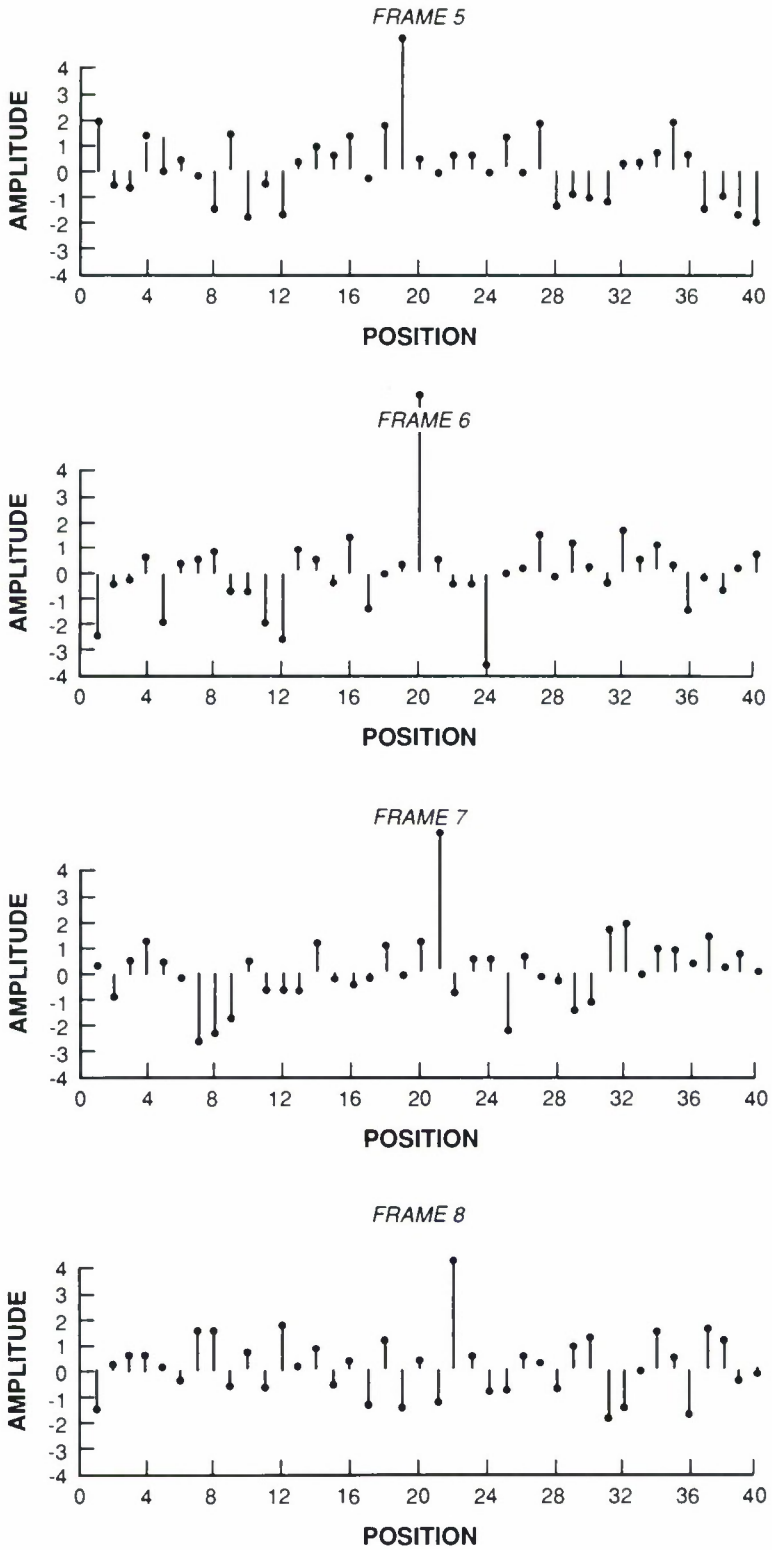


Figure 2-7 continued.

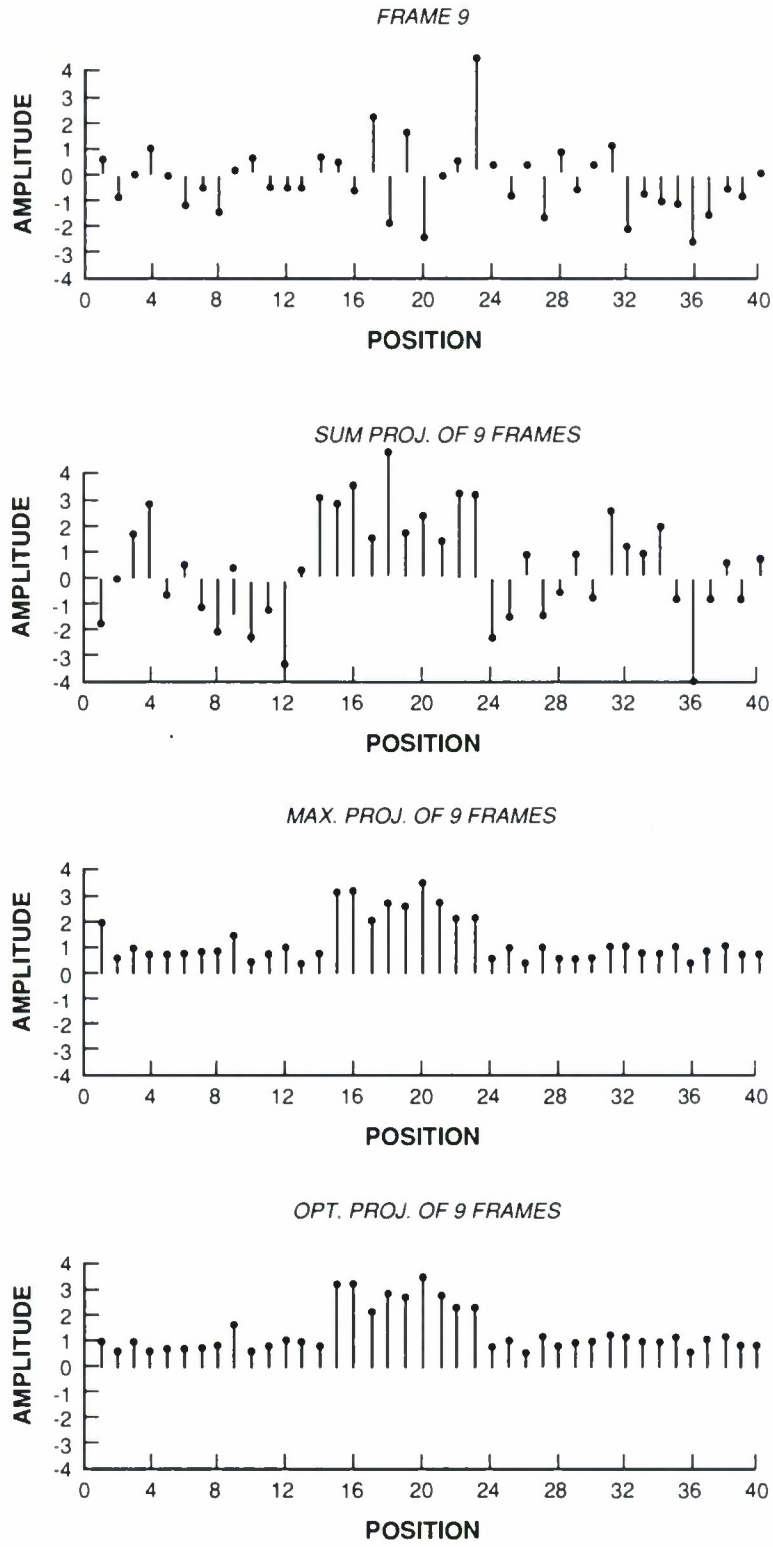


Figure 2-7 continued.

Note the clarity with which the projected signal track may be visually discerned in the maximum value projection compared to the conventional summation projection. Also note how close maximum value projection is to optimal projection for this example.

2.5 NORMALIZATION BY MEAN AND STANDARD DEVIATION

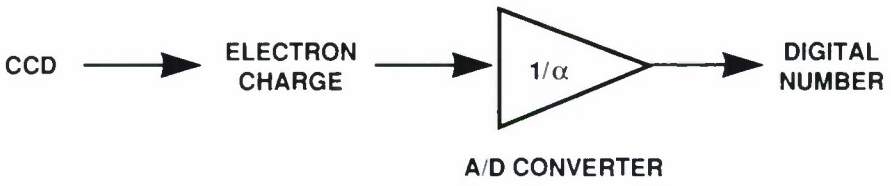
The pixel charge $r(x,t)$ is converted to a voltage and then to a digital number. The conversion factor is α, such that for an increment of α electrons in the pixel charge packet, the A/D output is incremented by 1 bit as shown in Figure 2-8. If $r'(x,t)$ is the A/D output for the pixel charge $r(x,t)$ then

$$r'(x,t) = r(x,t)/\alpha. \tag{2.23}$$

Equation (2.23) is not strictly true since the result of the division must be rounded up or down to an integer value. The difference between the integer value and the real value of the division is the quantization error of the converter. As long as this error is somewhat lower than the inherent noise of the pixel voltages, it can be ignored. As described in section 4.2.1, the maximum value projection along the time axis of the input data is taken

$$z(x) = \text{ maximum value of } [r'(x,t), t = 1, ..., N]. \tag{2.24}$$

To further reduce the amount of data to process, the maximum value projection will next be binary quantized with a constant threshold. The probability of each noise-only pixel exceeding the threshold must be equal. In an attempt to achieve this equality all maximum values have their means subtracted out and are divided by an estimate of their standard deviation, as shown in Figure 2-1.



117435-6

Figure 2-8. Electron charge to digital number conversion by the A/D converter.

While the mean is easily estimated for each location, estimation of the standard deviation is complicated by the presence of nonmoving objects (stars, galaxies, etc.) with jitter. Two methods are used to estimate the standard deviation. The first method assumes a Poisson (for the background illumination and dark current) plus Gaussian (for the circuit noise) model and derives

the standard deviation from the mean. The second method is a more heuristic approach which is sensitive to the presence of nonmoving objects. If the second method yields a standard deviation estimate that is substantially larger than the first method, the standard deviation of the second method is used. Otherwise, the estimate of the first method, which has much more accuracy and less variance, is used. If the first method alone were used to estimate the standard deviation, good detection performance would be possible for uniform backgrounds, but the presence of clutter would produce many false alarms. If the second method alone were used to estimate standard deviation, clutter would not produce false alarms, but the detection sensitivity for uniform backgrounds would be poor.

The mean is estimated by

$$\widehat{\text{mean}}(x) = \frac{1}{N-1}\left(\sum_{t=1}^{N} r'(x,t) - z(x)\right). \tag{2.25}$$

The estimate of the mean in (2.25) has the maximum value subtracted off as this maximum value may be the signal we are trying to detect. The resulting estimate is biased slightly lower than the actual value of the mean. Note that when $N = 2$ in (2.25), the estimate of the mean amounts to taking the minimum of two values at a given location.

After (2.25) is computed it is subtracted from the maximum value

$$z'(x) = z(x) - \widehat{\text{mean}}\,(x). \tag{2.26}$$

The standard deviation at a given location may be derived directly from the mean calculated in (2.25) if the major contributors to the pixel voltage at a given location are Poisson, Gaussian, and signal sources. The mean value of the charge is α times larger than the mean value as estimated in (2.25) because of the A/D converter

$$\text{mean } \widehat{\text{value}} \text{ charge } (x) = \alpha \cdot \widehat{\text{mean}}\,(x). \tag{2.27}$$

Since the Gaussian and signal sources do not contribute to the $\widehat{\text{mean}}\,(x)$, the charge mean value in (2.27) is the mean value of the Poisson component only. Therefore, the variance of the Poisson charge component is

$$\text{Poiss. } \widehat{\text{variance}}(x) = \text{mean } \widehat{\text{value}} \text{ charge } (x). \tag{2.28}$$

The variance of the Poisson component at the output of the A/D converter will approximately be

$$\text{Poiss. var. } \widehat{\text{at}} \text{ A/D out } (x) = \text{ Poiss. } \widehat{\text{variance}}(x)/\alpha^2. \tag{2.29}$$

Combining (2.27), (2.28), and (2.29), the Poisson variance at the output of the A/D converter is

$$\widehat{\text{Poiss. var. at A/D out}}\ (x) = \widehat{\text{mean}}\ (x)/\alpha. \tag{2.30}$$

The variance at the A/D output due to the Gaussian component is known since it is a fixed constant, unchanging with time and constant with pixel position

$$\text{Gauss. var. at A/D out}\ (x) = \sigma^2/\alpha^2 \tag{2.31}$$

where σ is the standard deviation of the circuit noise as defined in (2.3). Since the Poisson and Gaussian components are independent random processes, the total variance at the A/D output is simply the sum of the variances of each component. Therefore, using (2.30) and (2.31), the total standard deviation of the A/D output is

$$\widehat{\text{stdv}}_1(x) = \sqrt{\widehat{\text{mean}}(x)/\alpha + \sigma^2/\alpha^2}. \tag{2.32}$$

If a nonmoving object contributes a significant portion to the pixel voltage, then the standard deviation will usually be much higher than that predicted by the Poisson and Gaussian models. To avoid the error this creates, a second method of estimating standard deviation is as follows

$$\begin{aligned}\widehat{\text{stdv}}_2(x) \quad = \quad & 0.5 \cdot (\ \max_2[r'(x,t), t = 1, ..., N] - \\ & \min[\ \widehat{\text{mean}}\ (x-1),\ \widehat{\text{mean}}(x),\ \widehat{\text{mean}}\ (x+1)])\end{aligned} \tag{2.33}$$

where $\max_2$ denotes the second highest value of the set of $r'(x,t)$ and min denotes the minimum of the three estimated means. Expression (2.33) has experimentally been found to work well for a wide variety of looks and frame numbers from 2 to 16. Its rationale is as follows.

If the pixel at location x has statistics that fit the Poisson and Gaussian model, the implication is that neighboring pixels will have similar statistics. Therefore, the minimum of the mean values of the 3-pixel window will be close to the true mean at location x. The difference between the second highest value and the mean of a population is an estimate of the standard deviation. This method is somewhat noisier but more computationally efficient than the more conventional approach toward finding standard deviation in which the difference between the sum of squares and the squared sum is taken by the following.

$$\widehat{\text{stdv}}_{conven}(x) = \sqrt{\frac{\sum_{t=1}^{N} r'(x,t)^2 - z(x)^2 - (1/N)\left(\sum_{t=1}^{N} r'(x,t) - z(x)\right)^2}{N-1}}. \tag{2.34}$$

Nonetheless, the method of (2.34) works well in normalizing the projection if the number of frames N is greater than 7. If the number of frames is less than 7 there are not enough temporal samples in a given location to get a good estimate of the standard deviation, and the method of (2.33), which uses information about samples in neighboring locations, must be used to adequately normalize

the projection to prevent false alarms. For the case of a uniform background, (2.33) will simply be a less accurate estimate of the standard deviation of the pixel at location x than (2.32). Since moving objects (at high SNR) only affect maximum values, they will have no influence on (2.33).

On the other hand, if a nonmoving object is at location x, the statistics between location x and its neighbors will be quite different. In this case it is likely that the mean value of a neighboring pixel will be far less than the mean value at x, therefore (2.33) will yield a large value.

If stdv_2 of (2.33) is twice as great as stdv_1 of (2.32), then the estimate stdv_2 is used to normalize $z'(x)$ of (2.26)

$$z''(x) = z'(x)/\ \text{stdv}_2, \tag{2.35}$$

otherwise,

$$z''(x) = z'(x)/\ \text{stdv}_1. \tag{2.36}$$

The array $z''(x)$ is binary quantized and sent to a binary streak detector.

The method of (2.33) for obtaining standard deviation estimates is heuristic and therefore must be validated experimentally. This has been done, as discussed in section 3.6.

2.6 PAIRWISE BINARY QUANTIZATION

In single sample binary quantization, a "1" is assigned to a location x if the sample voltage at that location exceeds a threshold. While this is computationally efficient, a detection loss results which is a function of the threshold setting and the signal SNR. This detection loss may be significantly decreased by pairwise binary quantization.

Pairwise sums of the sample values are found about each location x. If any of these pairwise sums exceeds a threshold, then a "1" is assigned to location x. A "0" is assigned to location x otherwise. More precisely,

$$\begin{aligned} b(x) = 1 \qquad & \text{if } z''(x) + z''(x+1) > T \\ & \text{or } z''(x) + z''(x+1+nrow) > T \\ & \text{or } z''(x) + z''(x+nrow) > T \\ & \text{or } z''(x) + z''(x-1+nrow) > T \\ b(x) = 0 \qquad & \text{otherwise} \end{aligned} \tag{2.37}$$

where $nrow$ is the number of elements in a row of the image. If a streak goes through location x the signal will be present in both samples of at least one of the pairs of (2.37). The pairwise

sum will have an SNR 3 dB higher than that of a single sample. Therefore, for a fixed false alarm probability, the probability that the x location will be assigned a "1" when a signal is present will be higher than for single sample binary quantization. Roughly a 2-dB (20 percent in amplitude) increase in sensitivity results from using the binary quantization technique of (2.37) versus single sample thresholding for one percent of the samples set to "1." Because of the regularity of the operations in (2.37), it is easily implemented in hardware.

The total number of "1" pixels in the binary image has a large impact on the processing time of all subsequent processing as well as the detection sensitivity of the algorithm. To maximize sensitivity given processing time constraints, the binary threshold is set to pass a selected number of "1" pixels with the aid of a histogram. In this manner the number of "1" pixels retained for detection processing is as high possible while not exceeding the limit imposed by post-MTI processing time constraints.

2.7 SUMMARY OF MTI STEPS

In this section the six steps of the MTI filter are briefly summarized.

1. Find the maximum value projection of the frames

$$z(x) = \text{ maximum value of } [r'(x,t), t = 1, ..., N]. \tag{2.38}$$

2. Find second highest value of the frames

$$\max_2(x) = \text{ second highest value of } [r'(x,t), t = 1, ..., N]. \tag{2.39}$$

3. Find the mean value of the frames without the maximum

$$\widehat{\text{mean}}(x) = \frac{1}{N-1}\left(\sum_{t=1}^{N} r'(x,t) - z(x)\right). \tag{2.40}$$

4. Subtract off the mean value from the maximum value

$$z'(x) = z(x) - \widehat{\text{mean}}\,(x). \tag{2.41}$$

5. Divide the maximum value by the estimated standard deviation. First find the standard deviation via method 1

$$\widehat{\text{stdv}}_1(x) = \sqrt{\widehat{\text{mean}}(x)/\alpha + \sigma^2/\alpha^2}. \tag{2.42}$$

Then find the standard deviation via method 2

$$\widehat{\text{stdv}_2}(x) = 0.5 \cdot (\max_2[r'(x,t), t = 1, ..., N] - \min[\widehat{\text{mean}}\ (x-1),\ \widehat{\text{mean}}\ (x), \widehat{\text{mean}}(x+1)]). \tag{2.43}$$

If $\text{stdv}_2(x)$ is greater than $2 \cdot \text{stdv}_1(x)$

$$z''(x) = z'(x)/\ \text{stdv}_2 \tag{2.44}$$

otherwise,

$$z''(x) = z'(x)/\ \text{stdv}_1. \tag{2.45}$$

(6) Binary quantize the normalized MTI output

$$\begin{aligned} b(x) = 1 \qquad & \text{if } z''(x) + z''(x+1) > T \\ & \text{or } z''(x) + z''(x+1+nrow) > T \\ & \text{or } z''(x) + z''(x+nrow) > T \\ & \text{or } z''(x) + z''(x-1+nrow) > T \\ b(x) = 0 \qquad & \text{otherwise.} \end{aligned} \tag{2.46}$$

In Figure 1-2 are shown the first and last frames of a 16-frame set of 420 x 420-pixel CCD data, in which a fast and a slow satellite are present. Figure 1-2 also shows the binary quantized MTI output and the resultant detected streaks.

2.8 DETECTION OF OBJECTS AMID MOVING CLUTTER

Assume the telescope is not in sidereal track, so that the background clutter (stars, etc.) appears to be moving. This situation may arise if the object is being tracked by the telescope, so that the object appears stationary on the focal plane. Filip [22] has suggested how the MTI algorithm may be slightly modified to reject clutter with known, nonzero velocity while allowing legitimate moving objects to appear in the output.

As the data enter the MTI filter the pixel x location coordinate for each frame has an offset subtracted from it which is a function of the frame number t and the distance in the vertical and horizontal directions that the clutter moves from the beginning of one frame to the beginning of the next frame. The clutter movement is known because it is a function of the mount movement which is under control of the data processor. Denote the horizontal distance as d_h and the vertical distance as d_v measured in units of pixel widths. Then for frame t the offset is

$$\text{offset}(t) = nrow \cdot \text{ integer}[t \cdot d_v] + \text{ integer}[t \cdot d_h] \tag{2.47}$$

where the integer [ ] operation is defined as finding the closest integer to the continuous argument (i.e., no interpolation is used) and *nrow* is the number of pixels in a row on the CCD. The effect of this subtraction is that an object like a star will appear to be a stationary streak with as much as a half-pixel frame-to-frame jitter. The length of the star streak will be determined by the frame integration time. The MTI filter will suppress the clutter. The clutter suppression is illustrated in cartoon form in Figure 2-9 for both object track and sidereal track modes (which was described in sections 2.1 through 2.6).

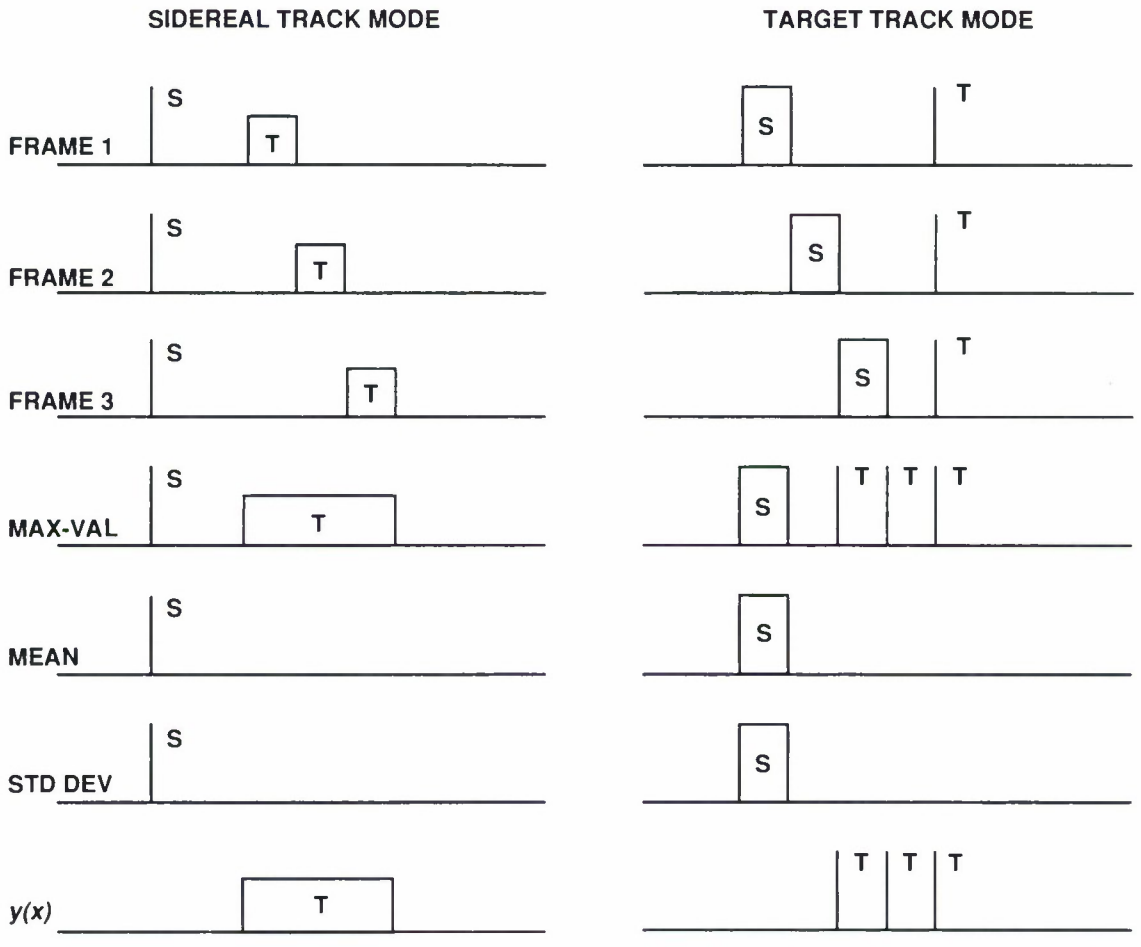


Figure 2-9. Clutter suppression for sidereal track and object track modes of the telescope mount. In the figure, "S" denotes a star and "T" denotes a moving object.

When the maximum value projection is found, the frame number of the maximum value at each location is saved in memory. After the binary quantization step, the frame number is used to reshift the address of each binary pixel back to where it would be if no offsets had been subtracted from the pixel coordinates at the MTI filter input. The post-MTI streak detection, discussed in section 3, may then take place. A block diagram of the algorithm is shown in Figure 2-10.

In Figure 2-11 are pictures showing intermediate steps in the algorithm using an actual streaked starfield with simulated targets. The data consist of 8 frames of 420 x 420-pixel CCD data taken from Lincoln's test site in Socorro, New Mexico.

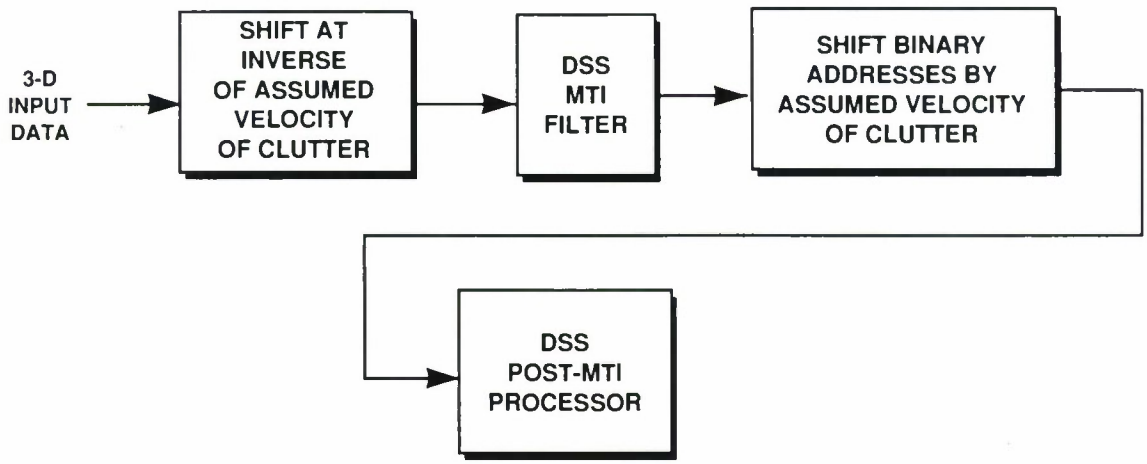


Figure 2-10. Block diagram of how MTI filter algorithm could be used to reject clutter moving at a known velocity.

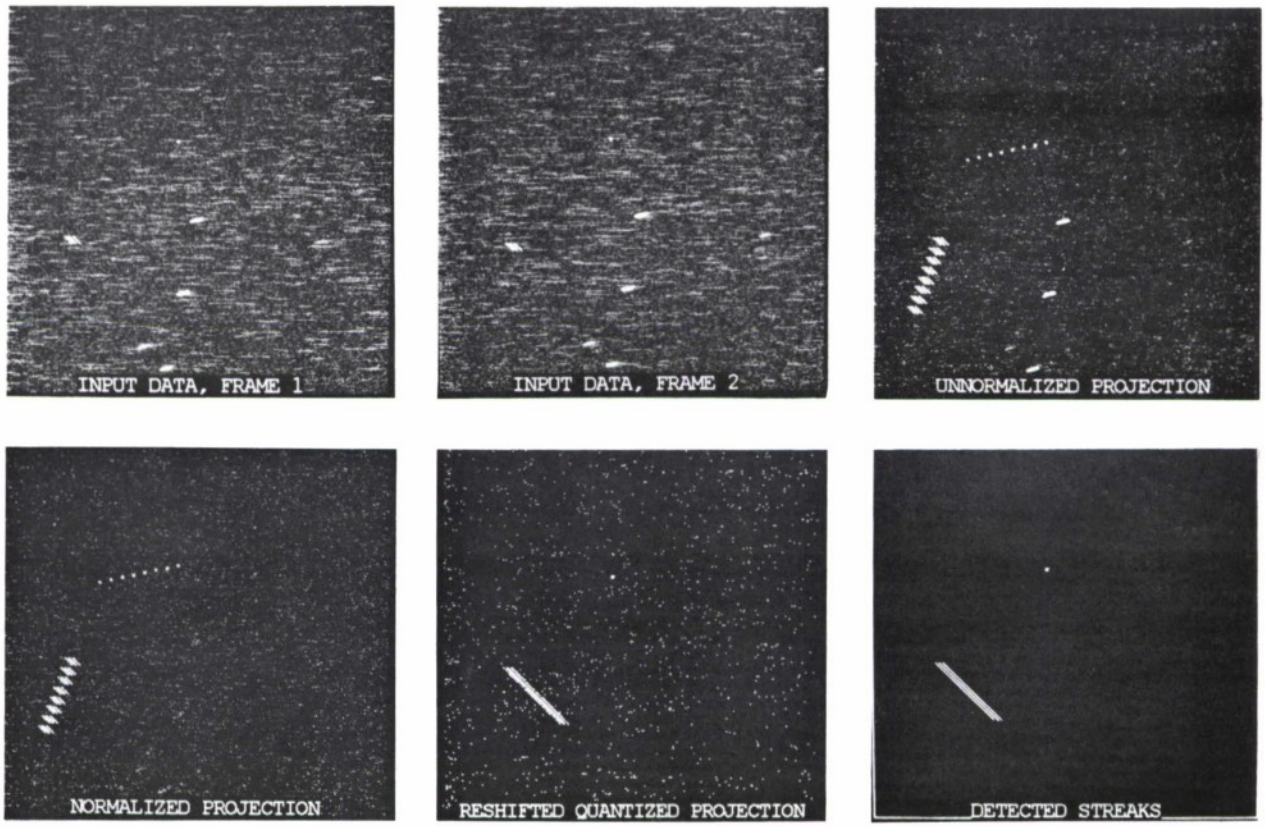


Figure 2-11. Intermediate steps in the block diagram of Figure 2-9 for actual data of 8 frames of 420 x 420 pixels with added simulated moving objects.

3. THE POST-MTI PROCESSOR

3.1 INTRODUCTION TO THE POST-MTI PROCESSOR

The post-MTI processor's input is a list of addresses where "1" pixels occurred in the binary quantized, normalized, maximum value projection of the MTI filter. Nominally, the binary threshold is set so that at most 1 percent of the 420 x 420 samples of the projection exceed the threshold, so that the list will consist of less than 2000 addresses. The post-MTI filter also has access to memories in the MTI processor which contain the full-precision, normalized maximum values and associated frame numbers during which the maximum value occurred.

As shown in Figure 3-1, the post-MTI processor performs the detection in a two-stage, "nominator-confirmer" procedure. The first stage searches for line segments of binary addresses, in effect, performing a binary matched filter search of all velocities. This procedure is computationally efficient but suboptimal in detection performance. The first stage serves to "nominate" streak paths which look like they may actually contain signal. The computationally expensive second stage "confirms" the nominations of the first stage by recalling the full-precision, normalized maximum values along the candidate paths, and summing these values (performing a full-precision velocity matched filter test). If the sum exceeds a threshold, the candidate path is declared legitimate, otherwise, the path is rejected. If the path is legitimate, the frame numbers along the path are recalled and used to estimate the speed of the moving object via linear regression. The angle of the object's movement is determined by the endpoints of the path.

In section 3.2 we discuss this "nominator-confirmer" philosophy, identical to multistage detection used in radar and sonar systems [18]. Section 3.3 is concerned with the theoretical basis and implementation of binary velocity filtering. "Analog" (or finely quantized) maximum likelihood streak path estimation is covered in section 3.4. The calculation of object speed via linear regression is explained in section 3.5. In section 3.6 measurements of metric accuracy and detection sensitivity of the algorithm are presented. The steps of the algorithm are listed in outline form in section 2.7.

3.2 NOMINATOR-CONFIRMER TWO-STAGE SIGNAL DETECTION

Suppose we are given many vectors, and we would like to select those vectors containing signal and reject those vectors containing solely noise. With known signal and noise parameters, the optimal detector may be derived. However, this function may be computationally too expensive to implement.

The computational expense of good detection algorithms can be mitigated by using crude detection algorithms to eliminate vectors from the data set that are obviously bad candidates for containing the signal. Once this is done the expensive detection procedure only has to be executed on the remaining vectors, which may be much smaller in number than the original data set. In

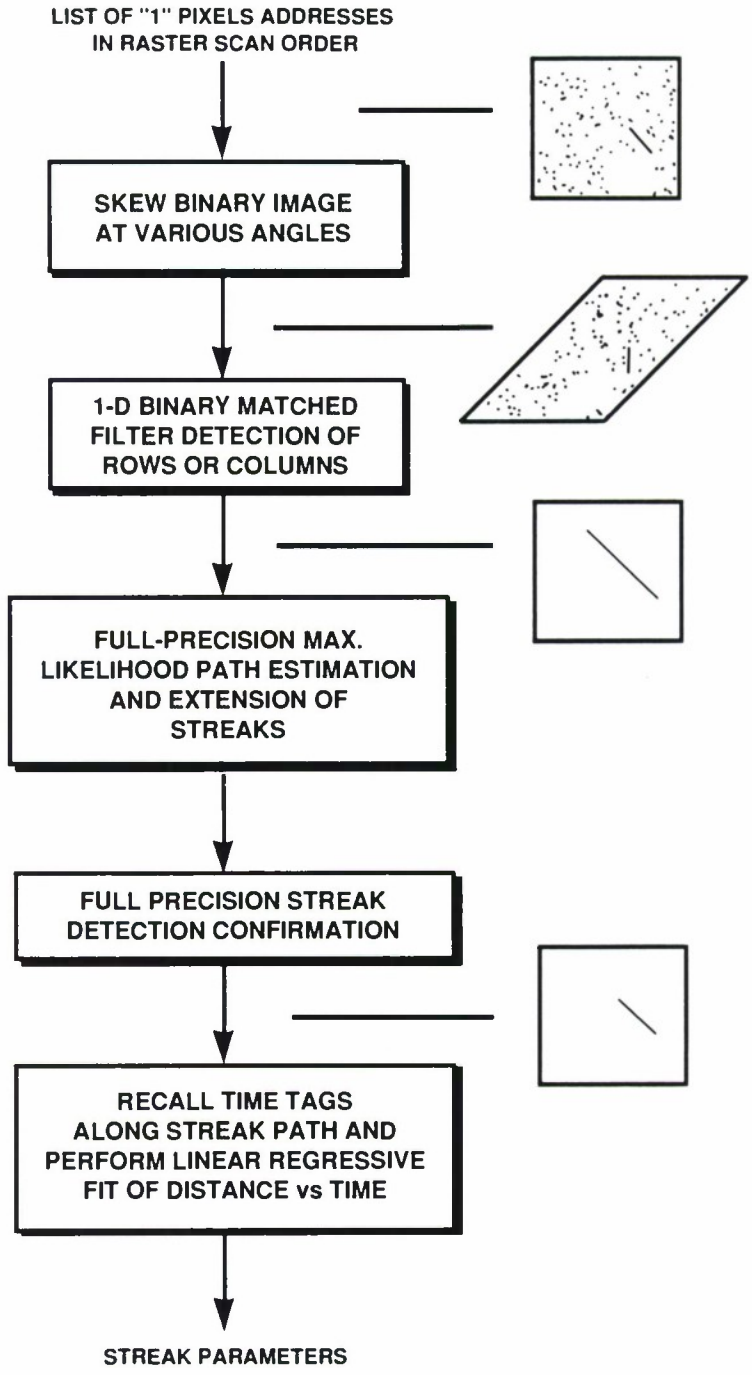


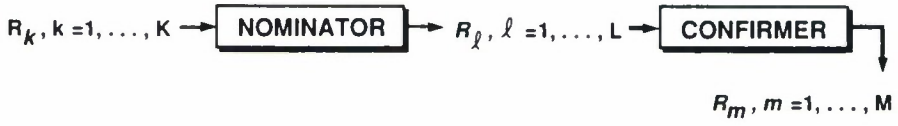
Figure 3-1. Block diagram of post-MTI processor.

this manner the bulk of the computational load may be transferred from the optimal detector to a crude detector.

3.2.1 Bounds and Optimality of the Nominator-Confirmer

In this section, bounds on the probability of detection p_d and probability of false alarm p_{fa} of the two-stage system are presented in terms of the individual nominator and confirmer p_d and p_{fa}. The nominator is a suboptimal, computationally cheap detector, while the confirmer is the optimal detector.

Assume we have a set of vectors $R_k, k = 1, ..., K$. The vector R_k is either a noise vector or a signal vector. In the nominator-confirmer scheme shown in Figure 3-2, the initial data set has K vectors. After nomination, the number of vectors is reduced to L, and, finally, after optimal selection, the number of remaining vectors (detections) is M. If L is, for instance, one-hundredth of K, then the confirmer will have to process only one-hundredth of the data that it would process if it were operating in isolation.



117435-19

Figure 3-2. Nominator-confirmer configuration for signal detection. $K > L > M$.

Lower and upper bounds on the probability of detection of the entire system are

$$\begin{aligned} p_d(\text{system}) &\geq p_d(\text{confirmer}) - (1 - p_d(\text{nominator})) \\ p_d(\text{system}) &\leq \min[p_d(\text{confirmer}), p_d(\text{nominator})]. \end{aligned} \tag{3.1}$$

The upper bound of (3.1) follows because the nominator and confirmer can only eliminate signal vectors from consideration, not add additional signal vectors. The lower bound results from the worst-case assumption that those signal vectors that the nominator rejects would have passed through the confirmer if it were operating in isolation with all K vectors as input. Under this worst-case assumption, p_d of the confirmer is reduced by the probability that the nominator will miss a signal vector $1 - p_d(\text{nominator})$. If the nominator and confirmer are at all similar in the way they choose signal vectors, this worst-case lower bound will never be attained.

The upper bound on the p_{fa} of the entire system is

$$p_{fa}(\text{system}) \leq \min[p_{fa}(\text{confirmer}), p_{fa}(\text{nominator})] \tag{3.2}$$

where the p_{fa} (confirmer) is referenced to the initial data set, not the reduced data set. The bound results because the nominator and confirmer can only reject noise vectors, not add additional noise vectors.

Using the bounds of (3.1) and (3.2) it is easy to show how nearly optimal performance can be obtained by this two-stage approach while achieving huge reductions in computational cost. Assume the optimal detector is the confirmer working in isolation, as shown in Figure 3-3, where all K data vectors are operated on directly by the confirmer. For example's sake, assume that the p_d of the optimal confirmer has been set to 0.99 while the p_{fa} is 10^{-6}. Now suppose we precede the confirmer stage with a nominator stage which has a p_d of 0.99 and p_{fa} of 10^{-2}. The nominator stage is suboptimal and therefore has a substantially higher p_{fa} than the confirmer stage for equal p_d. The confirmer of this system will operate on one-hundredth of the vectors it would operate on in the single-stage system. According to the bounds formula, the p_d of the two-stage system is ≥ 0.98 with $p_{fa} \leq 10^{-6}$. Thus, compared to the optimal detector, the worst-case performance loss of the two-stage system is only 1 percent in probability of detection. This slight reduction in detection sensitivity is well worth the possible hundred times decrease in computational cost (depending on the computational complexity of the nominator).

117435-20

Figure 3-3. Optimal detector: the confirmer in isolation.

3.2.2 Application of the Nominator-Confirmer to Streak Detection

The MTI filter projects the time sequence of data onto a single frame. The optimal streak detection method for the projected data is to feed the projected frame through a full-precision matched filter. This matched filter involves summing up projected data values along all possible streak paths (velocities) and comparing these sums to a detection threshold. Because there are so many paths to consider, the full-precision matched filter is difficult to implement [21]. Instead, the nominator-confirmer approach is employed. Each possible streak path in the projection is considered a vector which may contain signal. The frame is first binary quantized and fed through a binary matched filter, which is the nominator stage. The binary matched filter essentially counts how many "1" pixels are in each candidate path or vector. If the count is greater than a threshold which is a monotonically increasing function of the path length, a streak is declared present, and the full-precision amplitude data associated with the streak path is retrieved. Next, all paths declared as having streaks present are confirmed or rejected by comparing the sum of the full-precision amplitude values along the path to a second threshold, constituting the optimal confirmer stage of the algorithm.

The detection performance of this two-stage system is within 0.5 dB of optimal. In fact, even if binary matched filtering were the only basis for detection, the detection loss, which has been well studied in the literature [19,20], would be approximately 2 dB (under Gaussian noise assumptions).

3.3 TWO-DIMENSIONAL BINARY VELOCITY FILTERING

The binary matched filter is computationally efficient primarily because it involves operations only on "1" sample locations which are nominally set to 1 percent of the total number of samples. Even with the number of "1" samples set to this low a percentage, the loss compared to unquantized matched filtering is only about 0.5 dB (assuming pairwise binary quantization is employed in the MTI filter). The structure of the binary matched filter is shown in Figure 3-1. The heart of the algorithm is a very computationally efficient one-dimensional matched filter algorithm which essentially can detect vertical or horizontal streaks. To detect streaks at arbitrary angles, the binary image is skewed at equally spaced angles. The result is that detectable portions of streaks at any angle become horizontal or vertical. The skewing procedure is particularly efficient (compared to rotation) because it involves only one addition per binary point and a modest sized table.

3.3.1 Angle Search

The binary matched filter search involves first skewing the binary image so that streaks become vertical or horizontal and then doing a one-dimensional binary matched filter search on the vertical columns or horizontal rows of "1" samples. This skewing procedure must be repeated for enough angles so that the probability will be high that sections of all streaks of interest become vertical or horizontal for one of the skew angles. The location of the "1" samples is stored in memory. These locations are then transformed to present skewed versions of the binary image, so that streaks are distorted to become vertical or horizontal. A range of 90 degrees in streak angle is covered by the following transformation of the "1" sample x,y location

$$\begin{aligned} x_{new} &= x_{old} + y_{old} \cdot f \\ y_{new} &= y_{old} \end{aligned} \tag{3.3}$$

where $f = \tan\theta$, $-45^0 \leq \theta < 45^0$. After the transformation vertical columns of "1" samples are fed into a computationally efficient one-dimensional binary matched filter streak detector (to be discussed later).

To cover the remaining 90 degrees of possible streak angle, the following transformation is used

$$y_{new} = y_{old} + x_{old} \cdot g$$

$$x_{new} = x_{old} \tag{3.4}$$

where $g = \tan(\theta - 90^0)$, $45^0 \leq \theta < 135^0$. Horizontal rows of transformed "1" samples are fed to the one-dimensional streak detector. In (3.3) and (3.4) fractional values for the new coordinates are rounded to the nearest integer.

Since the angle of the streak is unknown, the transformations of (3.3) and (3.4) and subsequent one-dimensional streak detection on rows or columns must be repeated for all possible angles. The number of computations required for the algorithm is directly proportional to the number of skew angles. Therefore it is important to use as few skew angles as possible while maintaining desired detection sensitivity. We have performed extensive Monte Carlo simulations to find the worst-case detection loss as a function of streak length and number of angles. The worst-case loss occurs when a streak's angle is halfway in between two of the skew angles. The loss is relative to the case where streak angle is coincident with a skew angle. In Figure 3-4 plots are shown for the case where the Gaussian-shaped point spread function of the telescope has a standard deviation equal to 0.2-pixel widths. In Figure 3-5 the same plots are shown for a wider point function where the standard deviation equals 1-pixel width. As might be expected, if the streaks are skinny (relative to the pixel width) the worst-case detection loss is far greater than when they are fat. Related analytical studies may be found in [21].

The algorithm uses 20 skew angles, which, as Figure 3-4 shows, are sufficient to detect skinny streaks from 2 to 40 samples in length at arbitrary angles, with a worst-case loss of 1.5 dB in sensitivity compared to using an infinite number of skew angles.

A look-up table may be used to find $y_{old} \cdot f$ in (3.3) and $x_{old} \cdot g$ in Equation (3.4), eliminating the need for any multiplies in the skewing procedure. For a 420 x 420-pixel CCD array, 420 look-up table entries are needed for each angle of image skew, implying that for 20 angles, 8400 look-up table entries are needed. However, by taking advantage of symmetries in the skewing procedure, the look-up table memory may be reduced by a factor of 4 to 2100 entries.

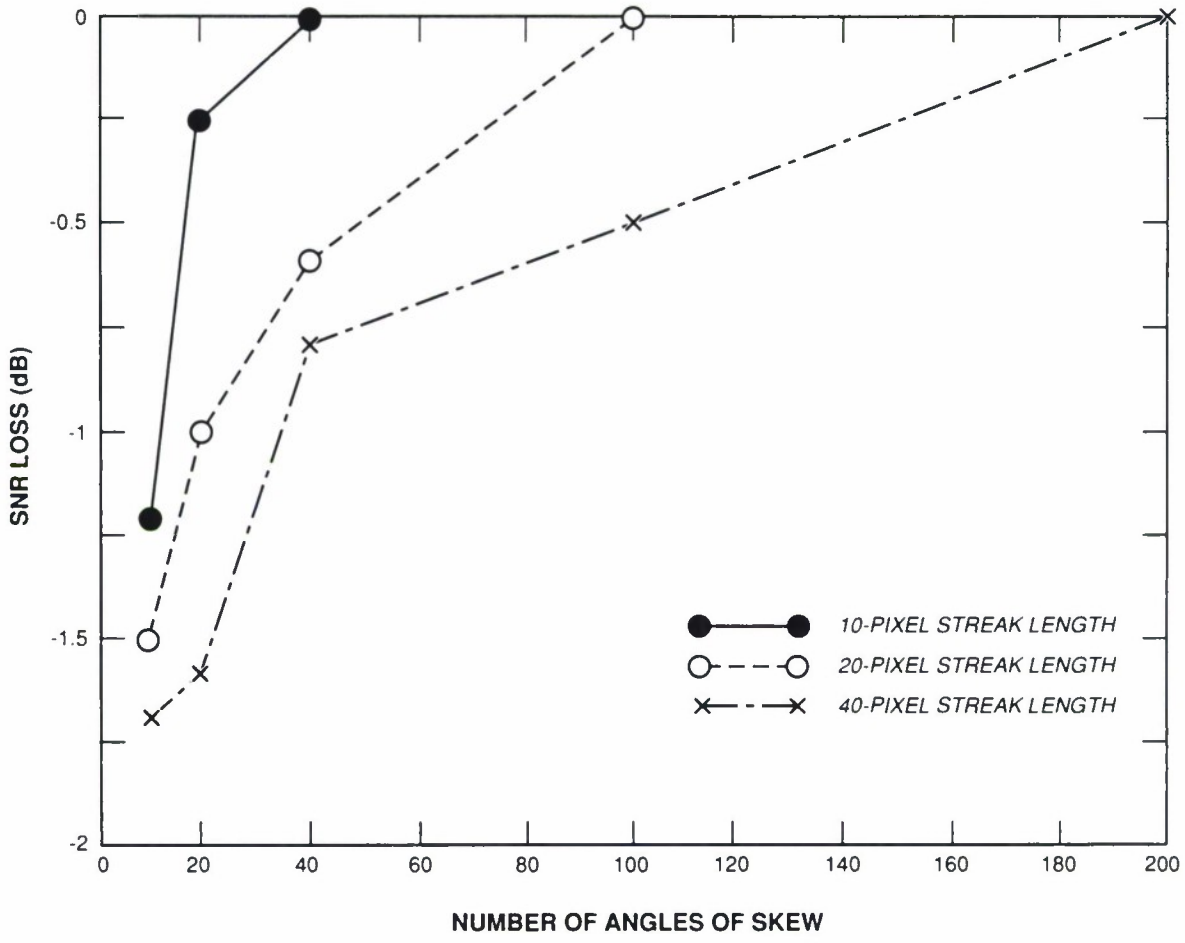
3.3.2 Speed Search

In the preceding section we described the skewing process used to search among a set of velocity vector angles. In this section computationally efficient method of searching through candidate speeds is described.

The candidate speed search is implemented via a one-dimensional binary matched filter search

$$\sum_{k=1}^{L} b(k) \overset{?}{>} T(L) \tag{3.5}$$

where $T(L)$ is a threshold which is a monotonically increasing function of streak length which is set to achieve as constant a false alarm rate as possible and the $b(k)$ are those binary samples in an



117435-7

Figure 3-4. Worst-case streak SNR loss in decibels as a function of the number of angles used in the skew. Point spread standard deviation equals 0.2 pixels (10 frames projected).

L sample candidate streak path. The threshold $T(L)$ may be determined by the classical binomial formula

$$p_{fa}(\text{streak}) = \sum_{i=T(L)+1}^{L} \frac{L!}{i!(L-i)!} p_{noise}^{i} (1 - p_{noise})^{L-i} \tag{3.6}$$

where p_{noise} is the probability that a pixel containing noise alone is "1," and $p_{fa}(\text{streak})$ is the probability of false alarm for detection of a streak in the L pixel path. If $T(L)$ is exceeded the full-precision maximum value projection voltages from the samples along the path are recalled

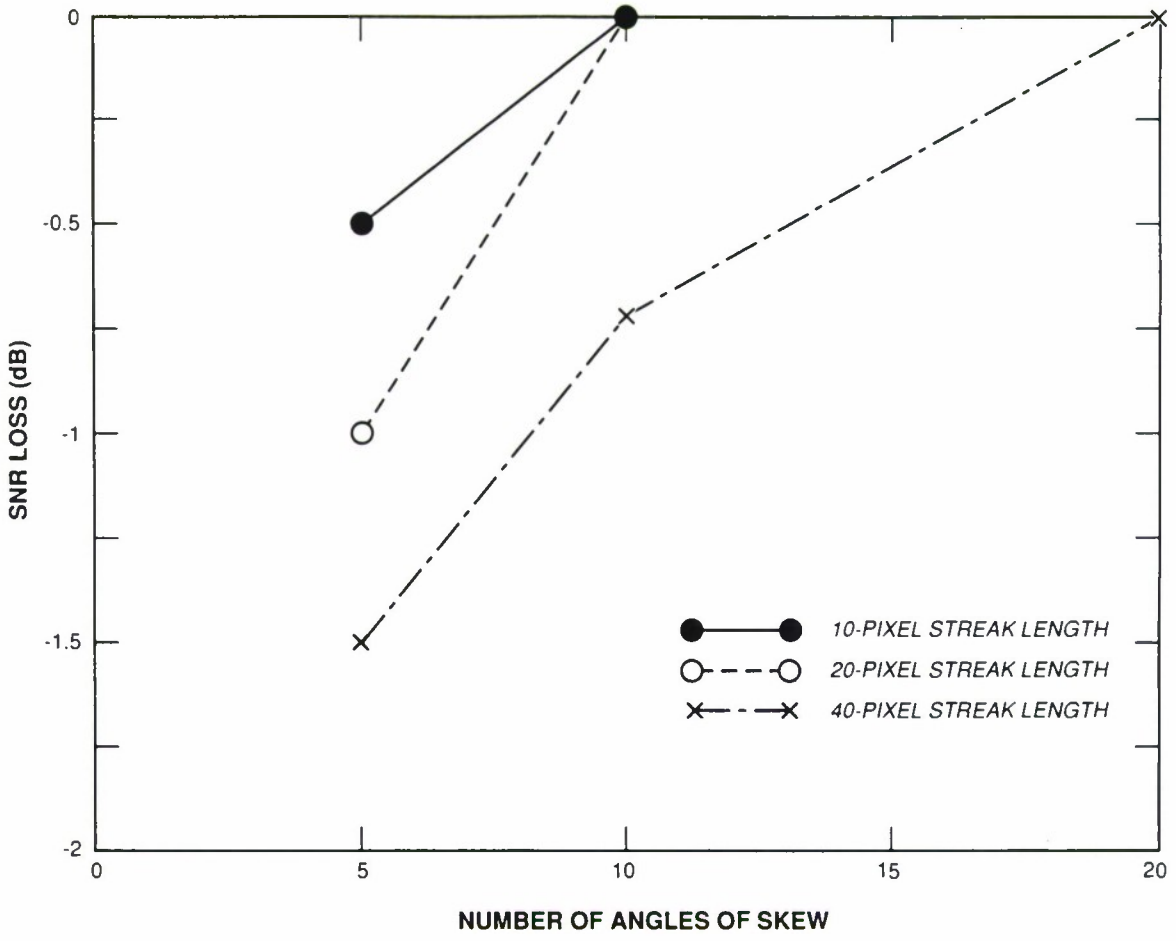


Figure 3-5. Worst-case streak SNR loss in decibels as a function of the number of angles used in the skew. Point spread standard deviation equals 1 pixel (10 frames projected).

and summed together, thereby implementing a full-precision matched filter for the assumed streak length and orientation (velocity vector). If this full-precision sum exceeds a second threshold, a streak is declared present. The full-precision matched filter confirms the binary detection and is necessary to reduce the probability of false alarm of shorter streaks, where even if all $b(k)$ in (3.5) are required to be "1" for detection, the false alarm rate is too high.

The test of (3.5) is applied only to candidate streak paths having "1" samples at their endpoints, because, as shown in Appendix A,

1. If any streak path will exceed the threshold, a streak path with "1" samples at

its endpoints will do so.

2. From a binary maximum likelihood point of view, the binary data are more likely to have been caused by streaks which happen to have "1" samples at their endpoints versus streaks which have a "0" sample at either endpoint.

The one-dimensional binary matched filter is computationally efficient, requiring M(M+1)/2 additions and comparisons where M is the number of "1" samples in the line. The following example illustrates the one-dimensional search algorithm.

Consider a row of eight samples:

$$00110001. \tag{3.7}$$

First, the binary data are coded in terms of the distance between neighboring "1" samples, except for the first "1" sample which is coded in terms of its distance from the start of the row. The resulting code for the sequence of (3.7) is 3 (position of the first "1" sample), 1 (distance between the first and second "1" sample), and 4 (distance between the second and third "1" sample). Denote the coded version of the sequence as

$$d_1, ..., d_M \tag{3.8}$$

where M is the number of "1" samples. Next, the distances of (3.8) are summed up in different ways to test for the presence of a streak

$$1 + \sum_{k=J}^{I} d_k \begin{cases} < T(I-J+2) & \text{declare streak present} \\ \text{otherwise} & \text{declare streak absent} \end{cases} \tag{3.9}$$

for $2 \leq J \leq M$, $J \leq I \leq M$. I and J are dummy variables. The term on the left-hand side of (3.9) is the length of the hypothetical streak (target speed), while I-J+2 is the number of "1" samples in the streak. The threshold function T(I-J+2) is chosen to give as constant a false alarm rate as possible, using the formulation of (3.6). For short streak lengths, the false alarm rate may be too high even with the threshold set so that all pixels in the path must be "1." For these cases, the false alarm rate can be lowered by testing the sum of the full-precision voltages along the path for the declared streaks.

As shown in Figure 3-1, the detector will output several candidate streaks. The final step of the algorithm involves picking the best candidate by picking the one whose path maximizes

$$\frac{1}{\sqrt{L}} \sum_{k=1}^{L} (z''(x) - \text{mean}) \tag{3.10}$$

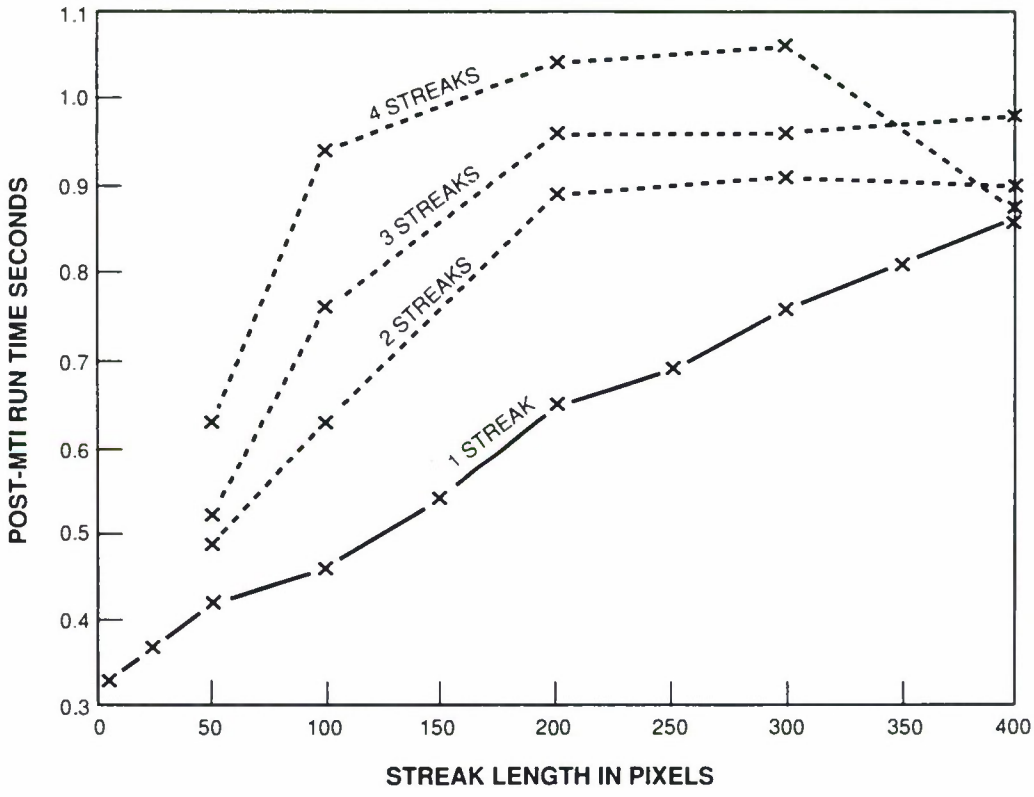
where the $z''(x)$ values in (3.10) are those normalized maximum value projected samples along the path of the streak (as defined in (2.35) and (2.36), section 2.5), and "mean" is the mean of the maximum value projected voltage for the noise only hypothesis. The rationale for using (3.10) is discussed in the next section.

The large majority of rows or columns may be excluded from consideration by a suboptimal preselection technique. As each skewed row or column is created, a count is kept of adjacent "1" samples. Two or more adjacent pairs of "1" samples must occur in the row or column before one-dimensional matched filtering is performed; otherwise, the signal is declared absent from the row or column. Only about a tenth of the rows or columns pass this test. On the other hand, if a streak is present in the line of data whose probability of detection is 0.95 without the preselection technique, then the probability of two or more adjacent "1" sample pairs is approximately 0.99 or so. The probability of streak false alarm is unaffected by this preselection technique while the probability of detection is, at worst, decreased by 1 percent. The number of computations needed to do the binary matched filtering is reduced by a factor of 10.

The number of computations needed for the binary velocity filter search is reasonable because a small percentage of the pixels have been set to "1." As the number of "1" samples increases, the computational cost increases, but the detection performance improves. Therefore, to obtain maximum sensitivity while staying within the limits of computational costs, a histogram is used to set the percentage of "1" samples as close as possible to an upper bound. This ensures that the processor will never be overloaded by an overabundance of "1" pixels even if many streaks are present, or there is clutter movement due to errors in the mount tracking system.

We have implemented the entire algorithm on a FORCE Motorola 68020 single board computer. With a 420 x 420-pixel image, the binary quantizer adjusted to pass one percent of the samples, and 20 angles of skew, all algorithm operations following the binary quantization are executed in 0.3 s real time with no streaks present. With streaks present, the run time increases, as shown in Figure 3-6, which contains benchmarks for the entire post-MTI process executed on a 68020 FORCE board for one to four streaks of 3-pixels width and different lengths. Note that even with four streaks of 400-pixels length, the total processing time is still on the order of a second. Also, note that the processing time decreases as the streak length is extended from 300 to 400 pixels for four streaks.

The processing time decrease is a by-product of the histogram process used to set the binary quantizer threshold. The top 1 percent of the 420 x 420 pixels with the largest amplitudes will be set to "1." These 1,764 pixels will be apportioned equally among four equal strength 400-pixel streaks, resulting in a binary image of four 400-pixel streaks of mostly single pixel width. However, when the 1,764 pixels are apportioned among four 300-pixel length streaks, there are enough pixels available to produce binary images for these streaks with multiple pixel widths for substantial sections of the streaks. Because the binary image of the streaks is wider for the 300-pixel streak length case than the 400-pixel streak length case, there are more candidate streak paths in the 300-pixel streak length case. The processor must spend extra time performing detection and likelihood tests for these extra candidate streak paths.



117435-21

Figure 3-6. Post-MTI run time versus streak length. Streak has 3-pixel width corresponding to point spread function of bright satellite.

The post-binary quantization operations are few in number compared to those required to perform the pre-binary quantization operations, maximum value projection and pairwise sample summation. However, the latter operations are readily amenable to parallel, fast special-purpose hardware because of their simplicity and regularity. On the other hand, the binary matched filter search is comparatively complicated and must be implemented on a general-purpose processor.

3.4 FULL-PRECISION MAXIMUM LIKELIHOOD PATH ESTIMATION

The pixel charges are converted to 12-bit quantities by an A/D converter. The quantization noise can be ignored because it is much less than the standard deviation of the circuit noise. Therefore, for ease of analysis, we can assume the representation of the pixel voltages to be continuous in the processor.

The detector may output several intersecting streak path candidates for each actual streak present on the focal plane, since the signal samples will be included in many hypothetical streak paths. This potentially increases the sum of the maximum values in those paths enough to exceed the detection thresholds. The path whose full-precision voltages produce the maximum likelihood probability is the one picked to represent the actual streak path.

Denote the full-precision, normalized maximum value projection samples as $z''(x)$. The probability density of $z''(x)$ under noise and signal hypotheses is complicated. To derive analytically tractable results, we shall approximate the noise as being additive, white Gaussian with known mean and the signal as being simply a dc offset. If a streak with constant SNR S (difference of the signal and noise means divided by the standard deviation of the noise), pixel length L, and start position at sample K, is present then the probability distribution of an individual sample in the set $z''(K), z''(K+1), ..., z''(K+L-1)$ is

$$p(z''(x)) = \frac{1}{\sqrt{2\pi}} e^{-(z''(x)-S-\text{mean})^2/2} \tag{3.11}$$

where "mean" is the mean of the noise. The probability distribution of a sample not containing the signal is

$$p(z''(x)) = \frac{1}{\sqrt{2\pi}} e^{-(z''(x)-\text{mean})^2/2}. \tag{3.12}$$

Using (3.11) and (3.12), the joint probability that all the data are caused by a streak of parameters K and L is

$$\text{p}(z''(1), z''(2), ...|\text{K}, \text{L}) = \prod_{\text{x=K}}^{\text{K+L-1}} \frac{1}{\sqrt{2\pi}} \text{e}^{-(z''(\text{x})-\text{S}-\text{mean})^2/2} \prod_{\text{j}\neq\text{x}} \frac{1}{\sqrt{2\pi}} \text{e}^{-(z''(\text{j})-\text{mean})^2/2}. \tag{3.13}$$

If S in (3.13) is unknown, the most likely value of S is

$$S = \frac{1}{L} \sum_{x=K}^{K+L-1} (z''(x) - \text{mean}). \tag{3.14}$$

If (3.14) is substituted into (3.13), then after some algebraic manipulation and taking logarithms, it can be shown that

$$\begin{aligned} &K, L \text{ that maximizes } p(z''(1), z''(2), ..|K, L) \\ &= K, L \text{ that maximizes } \frac{1}{\sqrt{L}} \sum_{x=K}^{K+L-1} (z''(x) - \text{mean}). \end{aligned} \tag{3.15}$$

Thus, among intersecting detected streaks, the one whose path maximizes (3.15) is chosen to represent the actual moving target.

While the measure of (3.15) is based on a crude Gaussian approximation, it has been found to produce accurate metric parameter estimates of real satellites. Some insight may be gained into the robustness of (3.15) by examining the case where (3.15) is used to pick one of two paths to represent the streak, a path of L pixels or the same path extended by one pixel to length L + 1. The longer path will be picked if

$$\frac{1}{\sqrt{L+1}} \sum_{x=1}^{L+1} (z(x) - \text{mean}) \geq \frac{1}{\sqrt{L}} \sum_{x=1}^{L} (z(x) - \text{mean}) . \tag{3.16}$$

Multiply both sides of (3.16) by $\sqrt{L+1}$

$$\sum_{x=1}^{L+1} (z(x) - \text{mean}) \geq \sqrt{1 + \frac{1}{L}} \sum_{x=1}^{L} (z(x) - \text{mean}) . \tag{3.17}$$

Next, using the approximation that $\sqrt{1+\delta} \approx 1 + \delta/2$ (3.17) may be rewritten as

$$\sum_{x=1}^{L+1} (z(x) - \text{mean}) \geq (1 + \frac{1}{2L}) \sum_{x=1}^{L} (z(x) - \text{mean}) . \tag{3.18}$$

Now let S denote the sum of the pixels in the path of length L so that

$$S = \sum_{x=1}^{L} (z(x) - \text{mean}) . \tag{3.19}$$

Combining (3.18) and (3.19)

$$S + (z(L+1) - \text{mean}) \geq (1 + \frac{1}{2L})S. \tag{3.20}$$

Finally, subtracting S from both sides of (3.20) yields

$$z(L+1) - \text{mean} \geq \frac{1}{2}(\frac{S}{L}). \tag{3.21}$$

The left-hand side of (3.21) is simply the maximum value minus the mean of the $L+1$ pixel, while the right-hand side is half of the average value of the L pixels common to both candidate paths. Therefore, the interpretation of (3.21) is that a path will be extended by a pixel if that pixel's

amplitude is greater than one-half of the average value of the pixels already in the path. This interpretation makes much sense heuristically.

If the angle of the streak doesn't coincide exactly with one of the 20 skew angles, the skewed streak will not align precisely with a row or column, and only a section of the actual streak will be detected for the skew angle. A line extension algorithm tries to extend the path of a detected streak in the skewed domain to adjacent columns based on the criterion of (3.15). If (3.15) is increased by the path extension, then the path is extended.

Occasionally a rotating rocket body may produce streak segments on the focal plane which are colinear but separated. Therefore, all estimated paths are checked for colinearity against each other. The time tags of those paths which are colinear are recalled and linear regression is used to see if the paths form a line in distance versus time space. If so, then the colinear paths are joined together and regarded as a single path.

Another potential anomaly is the occurrence of "glints" from dim rotating satellites. What is detected of the satellite are short, widely separated streak segments occurring in single frames as the satellite crosses the focal plane. Any single flash is not discernible from a cosmic ray hit. The algorithm looks at line paths formed by pairs of these segments. If three or more segments occur on a line path and the recalled time tags of the segments define a line in distance versus time space, then the segments are connected together. This procedure is robust against cosmic ray hits.

3.5 CALCULATION OF TARGET SPEED VIA LINEAR REGRESSION

While the detected streak on the projected plane gives us information about the angle of the target's movement, it may give erroneous information about target speed since the target path may not be detected in its entirety. This occurs, for example, when the target goes off the edge of the focal plane during a look, intersects clutter, or has a varying SNR. By recalling frame numbers during which the maximum values occurred along the path of the streak, the target's speed may be estimated. Denote the distance in pixels of the ith pixel along the path of the streak as d_i, and denote the frame number during which the ith pixel's maximum value occurred as t_i. Then a sequence of distance versus time pairs for an L-pixel streak is

$$\{d_i, t_i\}_{i=1}^{L}. \tag{3.22}$$

The equation of a line in the distance versus time coordinate space is

$$d = mt + b \tag{3.23}$$

where m is the slope of the line in pixels/frame and b is the d intercept. A popular approach toward finding the line which best fits noisy data is linear regression, in which the parameters m and b of the line are chosen to minimize

$$\sum_{i=1}^{L}(d_i - mt_i - b)^2. \tag{3.24}$$

The solution for m and b may be found by setting the partial derivatives of (3.24) with respect to m and b to zero. The resulting m and b are

$$\begin{aligned} m &= \frac{\sum d_i t_i - (\sum d_i \sum t_i)/L}{\sum t_i^2 - (\sum t_i)^2/L} \\ b &= \frac{1}{L}\sum d_i - m\frac{1}{L}\sum t_i. \end{aligned} \tag{3.25}$$

Linear regression is the maximum likelihood estimator in the situation where the errors between the fitted line and the data points are Gaussian distributed. Unfortunately, these errors are far from Gaussian in nature. The formulation of (3.25) is modified by three methods which empirically have been found to give better estimates of target speed:

1. End segment deweighting: The end segments of a streak are very prone to error. These segments may go off the focal plane, be too long for strong signals, or too short for weak signals. Therefore, they are deweighted by execution of a modified form of (3.25)

$$\begin{aligned} m &= \frac{\sum w_i d_i t_i - (\sum w_i d_i \sum w_i t_i)/(\sum w_i)}{\sum w_i t_i^2 - (\sum w_i t_i)^2/(\sum w_i)} \\ b &= \frac{1}{\sum w_i}\sum w_i d_i - m\frac{1}{\sum w_i}\sum w_i t_i \end{aligned} \tag{3.26}$$

   where the w_i are the weights assigned to each pair.

2. Amplitude weighting: Many rotating satellites will be fluctuating in brightness due to solar panels and other nonuniformities. Therefore, it makes sense to give more weight to those (d_i, t_i) pairs originating from high amplitude maximum values than those pairs due to low amplitude maximum values. The w_i in (3.26) are set higher for high amplitude maximum values and lower for low amplitude maximum values.

3. Median filtering: Occasionally, due to noise spikes, isolated values of t will be in error along the streak path. An example best illustrates how median filtering gets rid of these noise spikes.

   Consider the following sequence of (d_i, t_i) pairs [(1,1), (2,2), (3,3), (4,16), (5,5)] for a 5-pixel streak as shown in Figure 3-7.

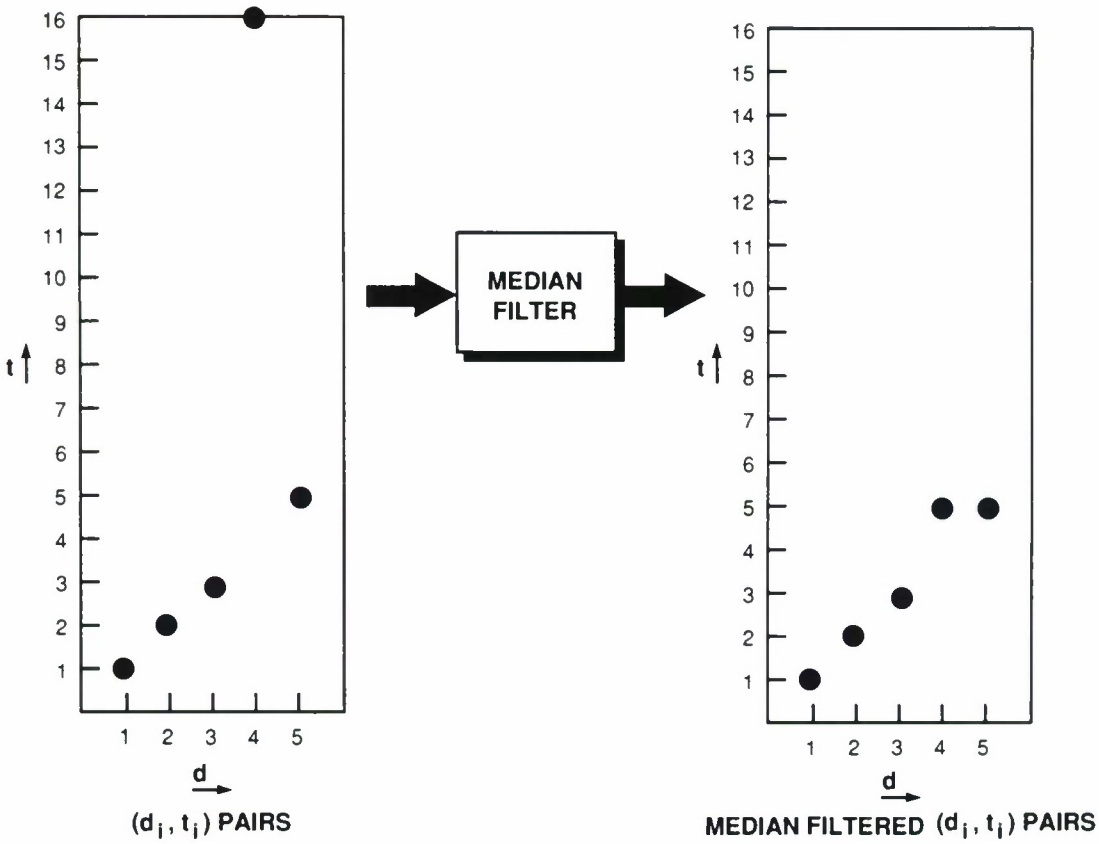


Figure 3-7. Example of elimination of noise spikes by median filtering.

The pair (4,16) was caused by a noisy pixel in frame 16 which happened to have a value larger than the signal containing pixel in frame 4. This sequence is median filtered by replacing the value of t_i in each pair by the median value of t_{i-1}, t_i, t_{i+1}. After this procedure, the t_i of the endpoints are clamped to differ from the t_i of the adjacent pair by at most two. In this manner isolated noise spikes will be drastically attenuated in their effect on linear regression.

3.6 ALGORITHM PERFORMANCE

3.6.1 Sensitivity

Detection sensitivity was evaluated by adding constant amplitude streaks of various SNRs to shuttered 9.6-s frame data and using Monte Carlo techniques to find the probability of detection. The streak probability of detection is 0.95 for a streak probability of false alarm of 10^{-10}. The

reason why this false alarm rate is set so low is that it is for a *single* streak. To approximate the false alarm rate for an entire look, the false alarm rate for the single streak must be multiplied by the total number of candidate streak paths being tested. The total number of streak paths is the number of starting positions (420 x 420) times the number of streak lengths (420) times the number of angles (20). When all these numbers are multiplied together, the approximate false alarm rate per look turns out to be 0.15. This number is probably at least an order of magnitude too high because all of the candidate streak paths overlap. Nonetheless, the point is that the single streak p_{fa} must be extremely low in order to keep the p_{fa} for the entire look low. In Table 3-1 is listed the required SNR (in amplitude, for a single, unprojected pixel containing the streak) required to achieve a streak $p_d = 0.95$ and streak $p_{fa}=10^{-10}$. The required SNR is a function of the number of frames and projected streak length.

TABLE 3-1.

Amplitude SNR Required for Streak p_d=0.95 and Streak p_{fa} =10^{-10}

Unprojected SNR_{amp} for Streak Detection				
Total Streak Length =	5	10	20	40
Number of frames = 2	6.4	4.3	3.6	2.9
Number of frames = 3	5.8	4.0	3.2	2.7
Number of frames = 5	6.0	4.0	3.4	2.8
Number of frames = 7	6.0	4.1	3.6	2.9
Number of frames = 10	6.0	4.1	3.7	3.2

Table 3-1 is useful in determining how many frames should be collected for a fixed stare time. For a fixed stare time, the projected length of a streak will be constant regardless of how the time is divided among the frames. As long as the object moves one or more pixels during the frame time, the number of object photons that hit a pixel is not affected by frame time. The shorter the frame time, the higher the single-pixel unprojected streak SNR will be, since less noise is accumulated

per pixel until we hit the lower limit on the frame exposure time, which is the time required for the object to move a pixel. Because the streak SNR required to produce the detection statistics of Table 3-1 is fairly constant as a function of frame number, the implication is that we should split up the stare time into as many frames as possible, as long as the object moves at least a pixel during the frame exposure time. Because endpoints of streaks on individual frames may be attenuated if there is appreciable signal in the same location on adjacent frames, it is good practice to choose the frame time so that the object produces a streak of at least a few pixels length in each frame.

The sensitivity of only the MTI section of the algorithm may be characterized in terms of the probability that a single binary pixel is "1" given that a signal with known SNR is present, for a binary threshold chosen to deliver a fixed single pixel probability of false alarm. Simulated streaks were added to 11 frames of shuttered CCD data which had equal components of Poisson and Gaussian noise, each being 45 electrons rms at the input to the A/D converter. A plot, shown in Figure 3-8, was made of the probability of a single binary pixel being "1" at the MTI filter output as a function of the unprojected streak single pixel SNR. The binary threshold was set so that the probability of a single binary pixel being "1" was 0.01, given that only noise was present at the pixel location.

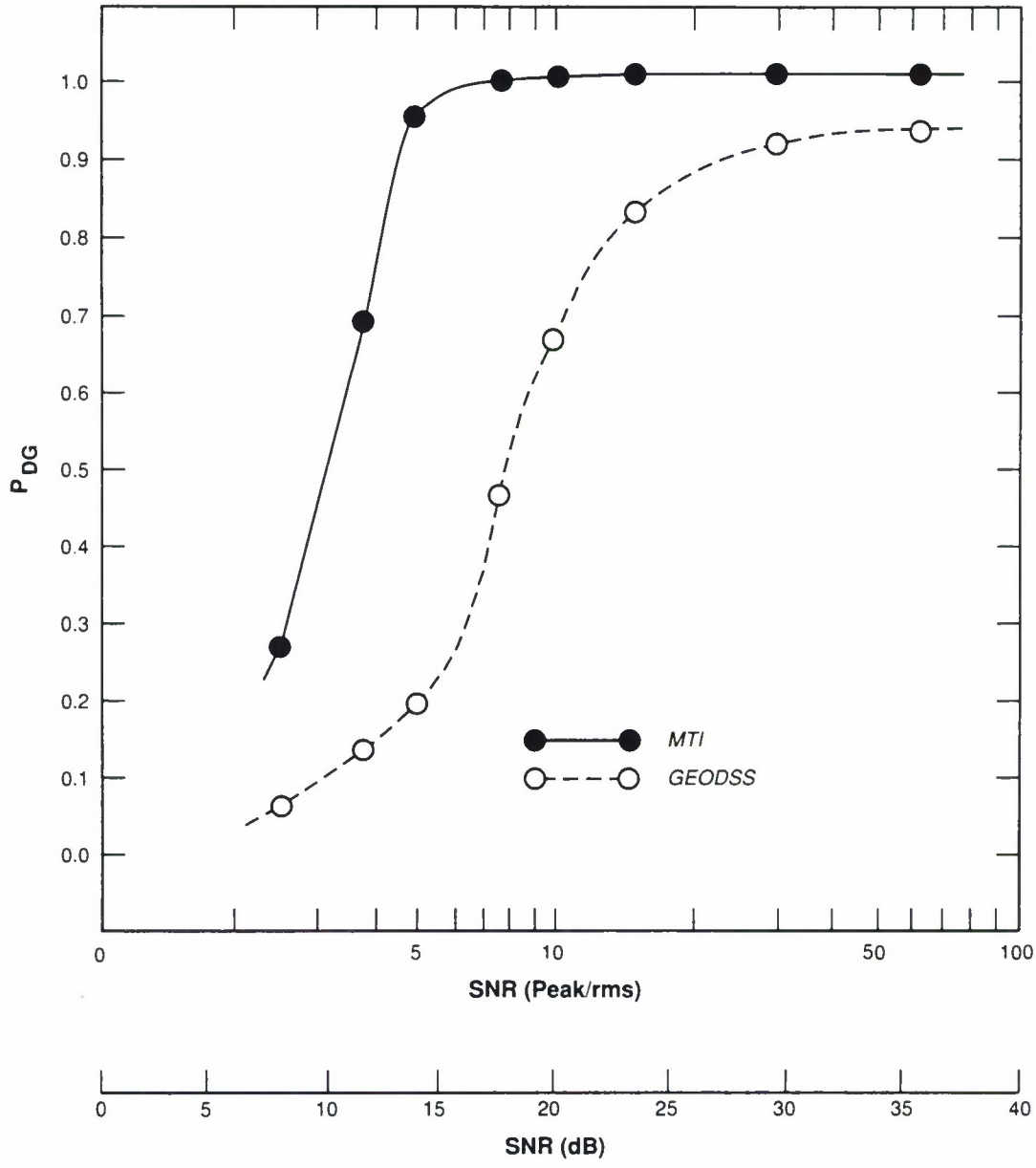
Also plotted in Figure 3-8 is the same detection sensitivity curve for the GEODSS[1] MTI filter, representing the current state-of-the-art. The GEODSS algorithm employs a constant false alarm rate (CFAR) technique of binary quantizing the signal at the output of the camera (a sliding window just prior to the pixel under question is used to estimate mean and variance). The early frames in the sequence are logically ORed to establish a background frame (B). This frame is complemented to form $\bar{B}$. Stellar rejection is then accomplished by forming the logical-AND, $D \bullet \bar{B}$, for each data frame D_i. All $D_i \bullet \bar{B}$ frames are then ORed together to form the OR frame. In these tests, the first 2 frames form the background, while the remaining 9 frames are data used to form the OR frame. The OR frame is passed through an isopixel cleanup to yield the final OR frame.

The MTI filter is roughly 6 dB or twice as sensitive as the GEODSS MTI filter.

3.6.2 False Alarm Rate

The complete satellite detection algorithm with both MTI and post-MTI processors has been tested on 120 x 120-pixel data taken at Socorro, New Mexico, with a 6-inch Celestron telescope. The results indicate that, for 2 to 11 frames per look, the algorithm will detect satellites with very low false alarm rates as long as the mount jitter is less than a pixel in extent. Most false alarms were due to anomalously large ETS mount drifts which will not happen if the mount is properly operating, and cosmic ray hits which may be eliminated if single-frame detections are ignored. The remaining situation causing false alarms occurs under low gain settings (320 electrons per A/D count) when the background is very bright (150,000 electrons output per CCD cell) and rapidly changing. Saturation effects in columns also cause false alarms. Ignoring the false alarms from

[1] Ground-based Electro-Optical Deep-Space Surveillance System.



117435-10

Figure 3-8. DSS MTI versus GEODSS MTI sensitivity.

abnormal mount drift, out of 442 looks, there were three false alarms due to cosmic ray hits and two due to a bright changing background.

3.6.3 Accuracy

In this section, the algorithm's estimation accuracy of a object's movement is discussed in terms of two components, angle and speed. Closed-form solutions for Cramer-Rao bounds on estimation accuracy for Gaussian-shaped objects with continuous sampling have been derived by Y. Chen [23].

In Figure 3-9 are some plots of the standard deviation of speed and angle errors as a function of amplitude SNR. In these cases the streak path is exactly vertical going through the centers of the pixel locations. Each error measurement was found by performing 20 runs, each consisting of processing 10 frames of Gaussian random noise plus signal. The point spread function was Gaussian with standard deviation of half a pixel width. Note that the estimation errors become negligible for SNRs in the neighborhood of 6 or so. For SNRs greater than 4 in amplitude, the rms speed error is within a factor of 2 of the Cramer-Rao bounds of [23]. However, the applicability of the bounds in [23] to our problem is questionable because of the assumptions of continuous spatial sampling, Gaussian distributed object shapes, pixels, and time exposures, and lack of access to frame time information.

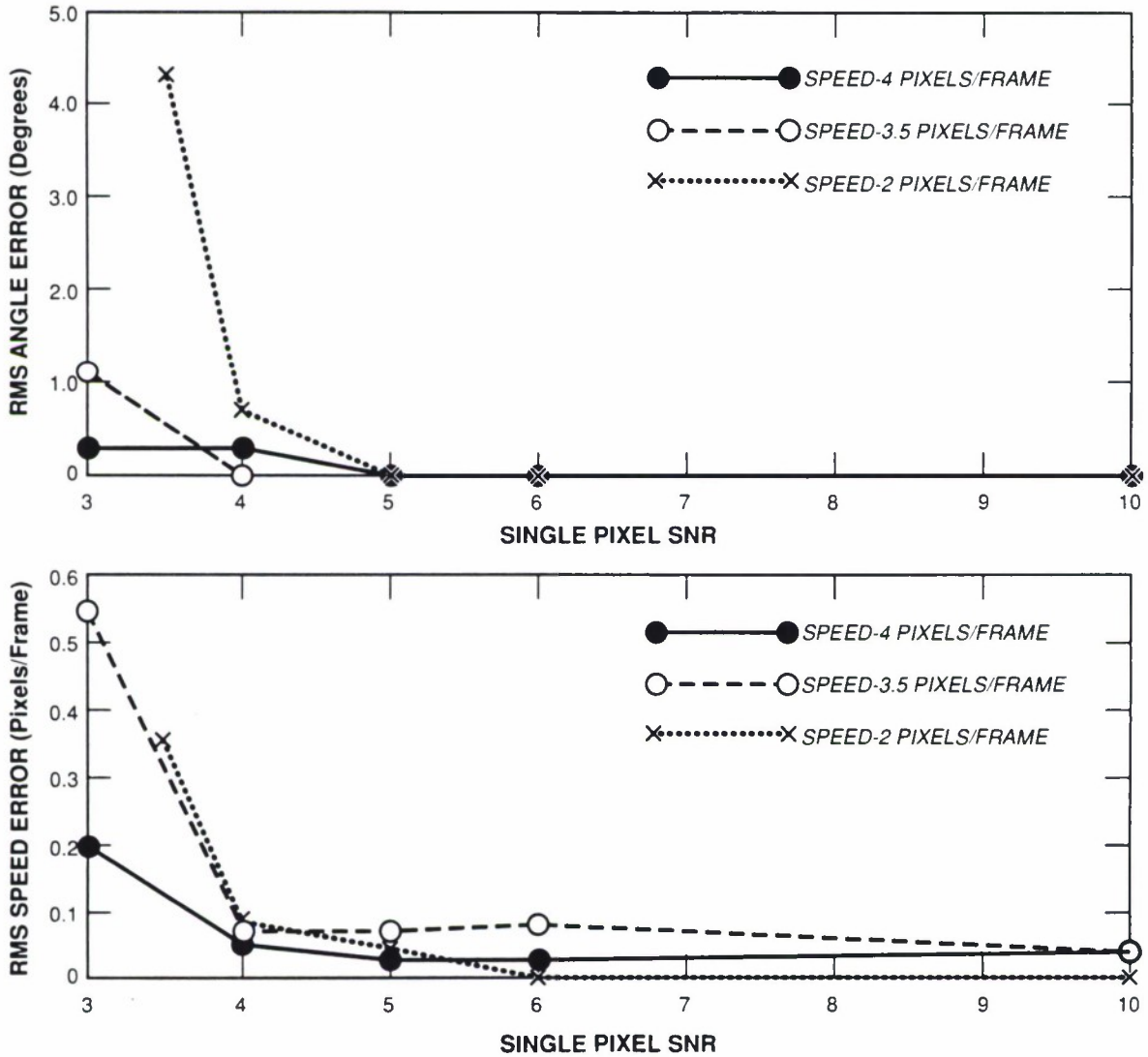
In Figure 3-10 we reran the simulations used to generate Figure 3-9 but with the streak now straddling pixels along its path rather than being centered. Note the significant decrease in accuracy estimates.

Even for high SNRs the accuracy in the angle estimate of the object's movement is limited by how precisely the path's endpoints can be determined. Because the algorithm doesn't use any kind of pixel interpolation, the endpoints can only be determined to the nearest pixel. Therefore, angle accuracy is a strong function of the streak position. If the streak path happens to be coincident with the pixel centers, there will be far less error compared to the case where the streak straddles pixels throughout its path. For streaks in arbitrary positions, the magnitude error of the endpoint position is approximately uniformly distributed from 0 to 0.5 pixels. The standard deviation of this error will then be $0.5/\sqrt{12}$ or 0.1443 pixels. The errors of both endpoints will add in an rms fashion, so that the angle error is

$$\text{stdv angle error} = \tan^{-1}\frac{\sqrt{.1443^2 + .1443^2}}{\text{streak length}}. \tag{3.27}$$

Errors in speed estimation have been measured using streaked starfields with 11 frames per look. These stars all had SNRs of 6 or greater. Results were as follows:

- At 2.75 pixels per frame, 11 frames, for 9 streaks, the rms speed error was 0.02 pixels per frame.
- At 9.00 pixels per frame, 11 frames, for 11 streaks, the rms speed error was 0.1 pixels per frame.



117435-11

Figure 3-9. Target velocity estimation errors with vertical streak, centered in pixels (10 frames).

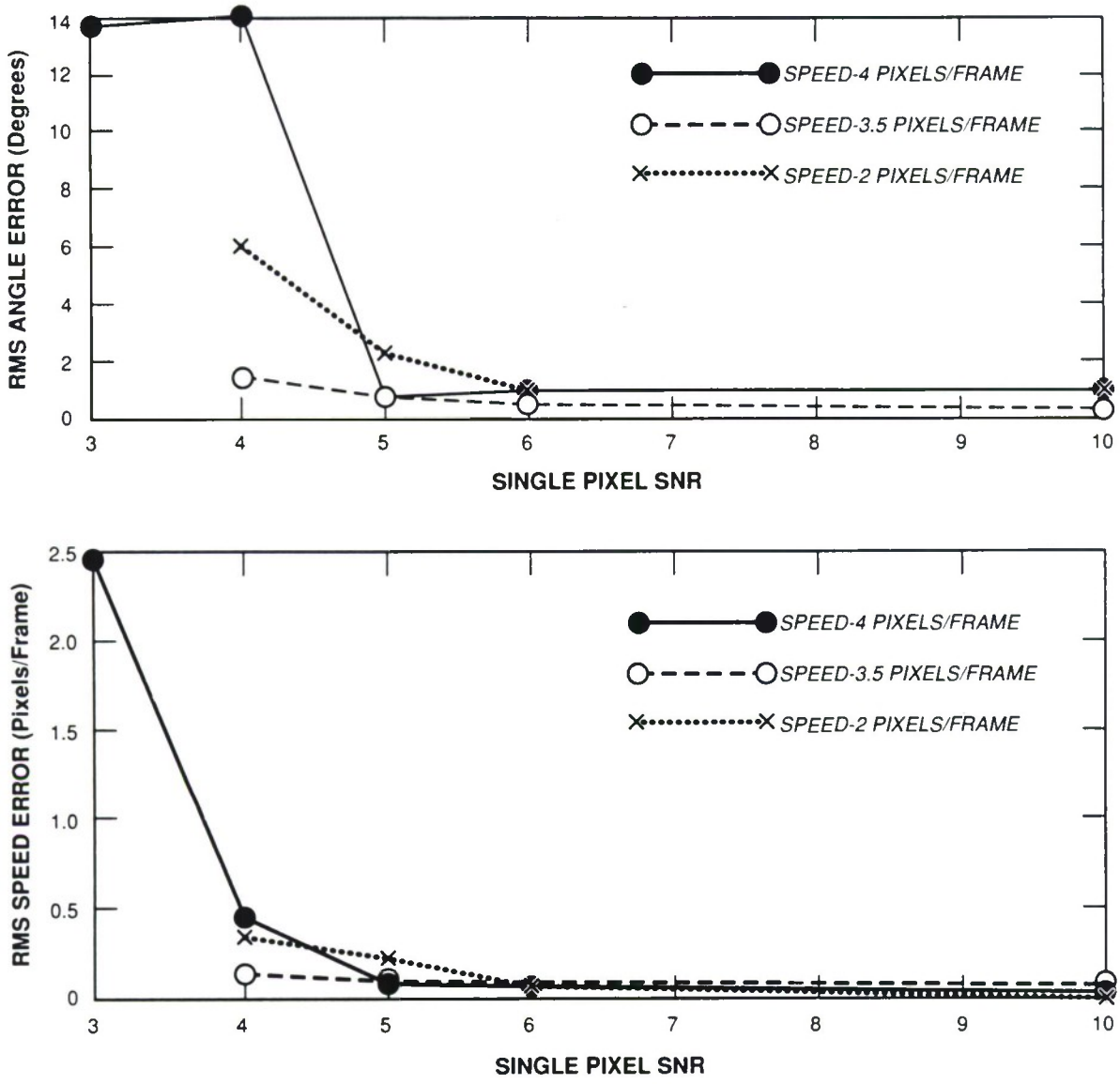


Figure 3-10. Target velocity estimation errors with vertical streak, straddling pixel boundaries (10 frames).

The streaks were nearly horizontal, and endpoints were estimated correctly to the nearest pixel in all cases. In several cases the streaks went off the focal plane during some of the frames.

For objects with SNRs greater than 6, a simple rule of thumb for predicting estimation accuracy is to assume that streak path endpoints may have as much as a 0.5-pixel magnitude error in position, and that object speed is determined by the length of the streak path (even though speed is determined by linear regression) and object angle is determined by the angle of the path.

Of course if an object fluctuates in brightness, the fluctuations may have an adverse effect on estimation accuracy. If the streak goes through a nebula or goes off the focal plane, estimation accuracy will also be degraded.

In Figure 3-11 are shown MTI outputs and the estimated streak path for an actual fluctuating object.

3.7 SUMMARY OF THE POST-MTI ALGORITHM

In this section, the steps of the algorithm are summarized.

1. Skew the binary image from the MTI processor at 20 angles.
2. Perform one-dimensional binary matched filter detection on rows or columns of the skewed images.
3. Use the full-precision maximum likelihood measure to pick the most likely streak path out of intersecting paths from the detector output.
4. Try extending paths using the full-precision maximum likelihood measure. Try connecting streak segments which are colinear.
5. Recall time tags along the streak paths and perform weighted linear regression to get speed estimates.

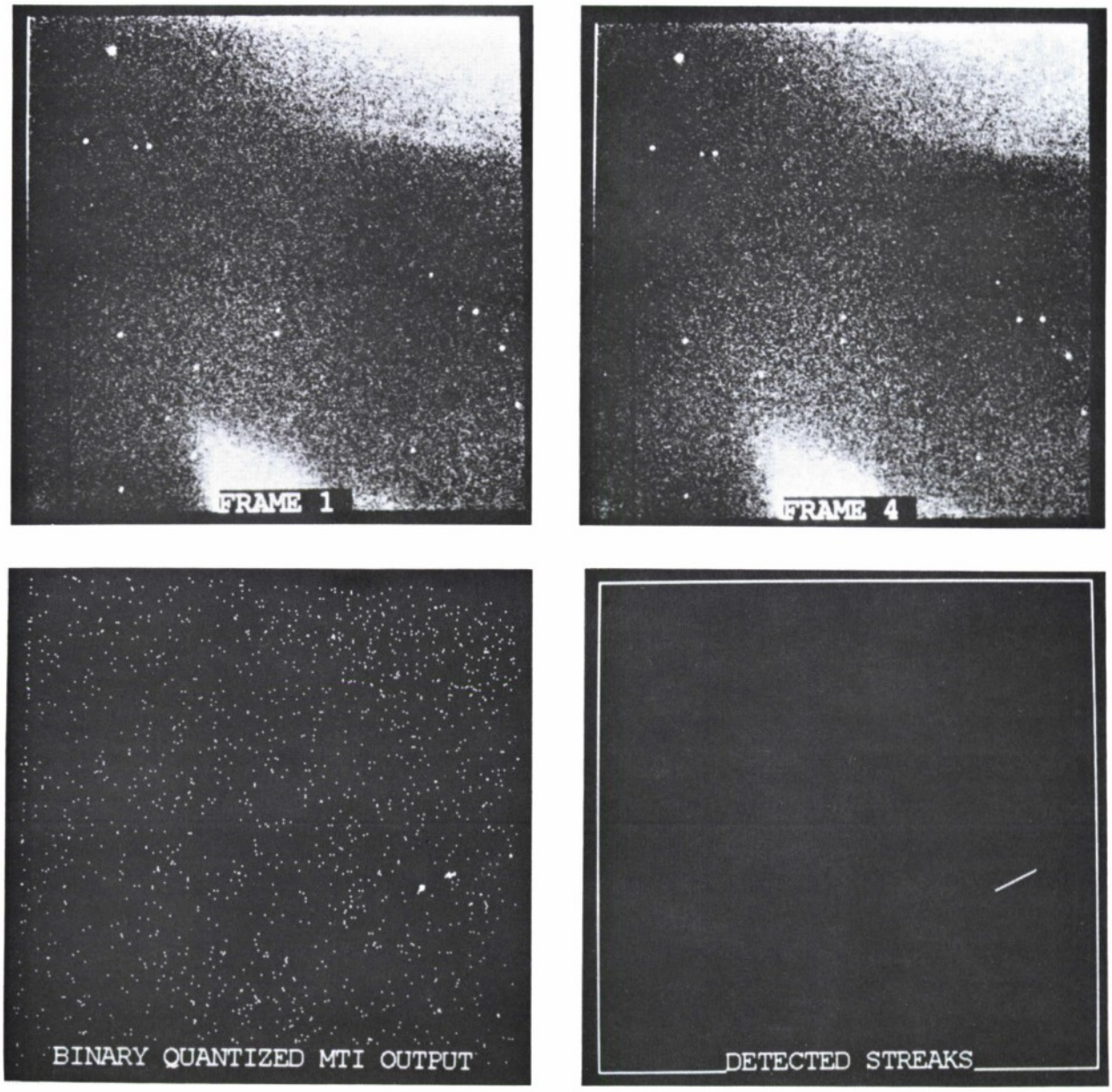


Figure 3-11. Streak path estimation of actual fluctuating object. Bright areas due to light leaks in the focal plane.

4. CONCLUSION

This report describes a suboptimal, computationally efficient algorithm for detecting small moving objects amid stationary clutter in a time sequence of data. Major techniques employed include sample normalization by mean and standard deviation to reduce clutter, maximum value projection of the data to reduce the dimensionality of the data in a computationally efficient manner with minimal detection loss, and a two-stage matched filter detector which performs a binary matched filter search followed by a full-precision matched filter test to confirm detections.

The algorithm has been tested on telescope CCD focal plane data and has been found to reliably detect satellites of sufficient SNR with very low false alarm rates amid astronomical clutter.

APPENDIX A

In this appendix it is shown that for a binary quantized image with a streak in it

- If any streak path will exceed the threshold of the binary matched filter, a streak path with "1" samples at its endpoints will do so.

- From a binary maximum likelihood point of view, the binary data are more likely to have been caused by streaks which happen to have "1" samples at their endpoints versus streaks which have a "0" sample at either endpoint.

The first assertion results from the fact that the binary matched filter threshold function $T(L)$ in (3.5) of section 3.3.2 is a monotonically increasing function of L, which is the length of the path in samples. Therefore, if the path is increased in length by a sample, the threshold will either stay the same or increase. If the newly included sample is "0," the threshold cannot be exceeded. However, if the newly included sample is a "1" sample, then the threshold may be exceeded. Therefore, if the sum of any binary samples in a hypothesized streak path can exceed $T(L)$, there must exist a path with "1" samples at its endpoints which can do so.

The second assertion is somewhat more difficult to prove. If a signal is present in the kth sample, then

$$\begin{aligned} p(b(k)=1) &= p_{sig} \\ p(b(k)=0) &= 1-p_{sig}. \end{aligned} \tag{A.1}$$

and if noise only is present in the kth sample, then

$$\begin{aligned} p(b(k)=1) &= p_{noise} \\ p(b(k)=0) &= 1-p_{noise}. \end{aligned} \tag{A.2}$$

Assume that samples with indices $K, K+1, ..., K+L-1$ contain a known streak of L samples length. The streak is parameterized by K and L. We wish to estimate K and L using the maximum likelihood approach.

For a given K and L, the probability of the binary data having occurred is

$$\begin{aligned} p(b(1),b(2),...|K,L) &= \prod_{k=K}^{K+L-1} (b(k)p_{sig} + (1-b(k))(1-p_{sig})) \\ &\quad \prod_{j\neq k} (b(k)p_{noise} + (1-b(k))(1-p_{noise}))\,. \end{aligned} \tag{A.3}$$

The estimated K and L are those values which maximize (A.3), which is a function of all the binary data. Alternatively, the logarithm of (A.3) may be maximized.

Suppose we are trying to decide which of two paths, of lengths L and $L+Z$, is more likely to have caused the binary data. The first path ends in "1" pixels, and the second path is simply the first path lengthened by Z samples which are all "0." Then from (A.3),

$$\log p(b(1), b(2), ...|K, L) \quad - \quad \log p(b(1), b(2), ...|K, L+Z) =$$
$$Z\left(\log\left(1 - p_{noise}\right) - \log\left(1 - p_{sig}\right)\right). \tag{A.4}$$

Since $p_{sig} > p_{noise}$, the term on the right-hand side of (A.4) is positive. Therefore, the candidate path ending in "1" samples is more likely to have caused the binary data. Similarly, if the path with "1" samples at the endpoints is decreased by Z "0" samples, the probability of the shortened path having caused the binary data is less. Therefore, the streak paths most likely to have caused the binary data have "1" samples at their endpoints.

APPENDIX B

In this appendix is the "C" code which implements the MTI algorithm. Its input is a raster scan ordered file of 16-bit data corresponding to the CCD pixel voltages. Subsequent frames in time are assumed to follow one another in the correct order. The program outputs a file containing addresses, time tags, and amplitude values of the "1" pixels of the binary quantized maximum value projection of the input data, other files useful for diagnosis as well as streak parameter estimation by the post-MTI processor.

The code for the post-MTI processor is not included because of its prohibitive length. A copy of this code may be obtained from Group 27 at MIT Lincoln Laboratory.

```
/* Performs the mti operation . Assumes frame is 420 by 420 */
/* outputs the intermediate results to files */
/* copyright (c) 1988 by MIT; all rights reserved.
            Developed at Lincoln Laboratory. */
#include<math.h>
#include<stdio.h>

short int buffer[177242];
short int max[176840];
short int max2[176400];
short int average[176400];
short int frame[176400];
int n_electrons,n_circuit,n_of_frames;

struct TargetType

    {

    short int X;

    short int Y;

    short int amp;
    };

struct TargetType TarDat[18001];


main()
    {

    int n,x,y,z,x_limit,y_limit,z_limit,index,status,size,offset,skip;
    int number_ones;

    int min,recstdv1,recstdv2,binary_quantize();
    int variance;
    FILE *fopen(),*input_file,*max2_file,*frame_file,*TarDat_file;
    FILE *mti_file;
    char name[200],filetype[20],disk[5],word[81];
    short int mti_root[256];
 /* generate the table of 1/sqrt */
    for(x=1;x<=255;++x)
```

```
        mti_root[x]=1024/sqrt(x+0.);
    mti_root[0]=1024;
/*

 Initialize the max, max2, average, and frame arrays

   */

   for(x=0;x<176400;++x)

        {
        max2[x]=0;
        average[x]=0;
        frame[x]=0;
        }
   for(x=0;x<176840;++x)
        {
        max[x]=0;
        }
/*

 set up output files for max2, TarDat.x,y,amplitude, and frame number
*/



   max2_file=fopen("max2.data","w");
   frame_file=fopen("frame.data","w");
   TarDat_file=fopen("TarDat.data","w");
   mti_file=fopen("mti.data","w");


/* Perform setting up procedure for inputting the ETS data */



   printf(" Input name of file\n");
   scanf("%s",name);
   input_file=fopen(name,"r");

/* Ask for other parameters */
```

```
    x_limit=420;
    y_limit=420;
    printf(" Input no. of 420 by 420 frames \n");
    scanf("%d",&z_limit);
    n_of_frames=z_limit;
    size=x_limit*y_limit;

    printf(" Input no. of electrons per A/D count ( 3 for hi gain )\n");
    scanf("%d",&n_electrons);
    printf(" Input circuit noise variance at A/D out \n");
    scanf("%d",&n_circuit);
/* Now input the frames, one at a time, and update the maximum,

   second maximum, and average buffers */

    for(z=0;z<z_limit;++z)
         {

         status=fread(buffer,2,size,input_file);
         printf(" Number of 16 bit words read, is %d\n",status);
         for(y=1;y<=y_limit;++y)

              for(x=1;x<=x_limit;++x)

                   {

                   index=(y-1)*x_limit+x-1;

                   if(buffer[index]>max2[index])
                        {

                        if(buffer[index]>max[index])
                             {

                             max2[index]=max[index];

                             max[index]=buffer[index];

                             frame[index]=z;
                             }

                        else max2[index]=buffer[index];
```

```
                    average[index]+=max2[index];
                    }

               else average[index]+=buffer[index];
               }

          }

/*   The maximum, max2, and average arrays have all been updated.

     Normalize the average array and subtract it off

     */

     n=z_limit-1;

     for(x=0;x<176400;++x)
          average[x]=average[x]/n;
 /* Subtract off the average value from the maximum value, and

     normalize by the standard deviation */

     for(x=1;x<176399;++x)
          {
          min=10000;
          for(n=(-1);n<=1;++n)
               {
               if(min>average[x+n])min=average[x+n];
               }
          if((max2[x]-min)>1)
                    recstdv2=2048/(max2[x]-min);
          else recstdv2=2000;
          variance=(average[x]/(n_electrons+0.))+n_circuit;
          if(variance<0)variance=0;
          if(variance>255)variance=255;
          recstdv1=mti_root[variance];
          max[x]=(max[x]-average[x]);
          max2[x]=max[x];
          if(recstdv2<(recstdv1/2))
               {
               max[x]=(max[x]*recstdv2)/64;
               }
```

```
        else
             {
             max[x]=(max[x]*recstdv1)/64;
             }
        }
    for(x=0;x<176400;++x)

        {
        if(max[x]<0)max[x]=0;
        if(max[x]>1999)
             {
             max[x]=2000;
             }
        }
/* Now OR the max value with the time value,as will be done in the MTI board*/
    for(x=0;x<176400;++x)
        {
        frame[x]=frame[x]<<12;
        frame[x]+=max[x];
        }


/* Output information corresponding to the frame during which the
   maximum value occured */
    fwrite(max2,2,176400,max2_file);
    fwrite(frame,2,176400,frame_file);
    fwrite(max,2,176400,mti_file);


/* Now binary quantize the output file , n is number of one pixels*/

    n=1760;
    number_ones=binary_quantize(n,x_limit,y_limit);

/* Output TarDat data */
    for(x=1;x<=number_ones+1;++x)
        {
        buffer[3*x-3]=TarDat[x].X;
        buffer[3*x-2]=TarDat[x].Y;
        buffer[3*x-1]=TarDat[x].amp;
        }
    fwrite(buffer,6,number_ones+1,TarDat_file);
```

```
    }

/*************************************************************************/

/* The following subroutine binary quantizes the array max */

int binary_quantize(ones,x_limit,y_limit)

int ones,x_limit,y_limit;

    {


    int x,y,sum,number_ones,index,maxsam,try,threshold;
    int quant_flag;
    int local;
    printf("input 1 for pairwise binary quantization, 0 for single\n");
    scanf("%d",&quant_flag);
    for(x=0;x<=4000;++x)buffer[x]=0;

    local=33;
       if(n_electrons>10)local=0;
    if(quant_flag==0)
       {
       for(x=0;x<176400;++x)
           {
           buffer[max[x]]+=1;
           average[x]=max[x];
           }
        }
    else
        {
        for(x=0;x<176400;++x)
           {
           maxsam=0;
           try=max[x+1+420];
           if(try>maxsam)maxsam=try;
           try=max[x+420];
           if(try>maxsam)maxsam=try;
           try=max[x-1+420];
           if(try>maxsam)maxsam=try;
           try=max[x+1];
```

```
            if(try>maxsam)maxsam=try;
            if(max[x]>local)
                    average[x]=maxsam+max[x];
            else average[x]=0;
            buffer[average[x]]+=1;
            }
        }
 /* Now find the threshold for binary quantization */

    sum=0;

    x=4000;

    do

         {

         --x;

         sum+=buffer[x];

         }

    while (sum<ones);

    number_ones=0;

    threshold=x;

    printf(" Input -1 for autothreshold, threshold value for manual\n");
    scanf("%d",&x);
    if(x!=(-1))threshold=x;
    printf(" The threshold is %d\n",threshold);
/* now binary quantize in a raster scan order */

    for(y=1;y<=y_limit;++y)

         for(x=1;x<=x_limit;++x)

              {

              index=(x-1)+(y-1)*420;
```

```
            if(average[index]>threshold)
                 {

                 ++number_ones;

                 TarDat[number_ones].X=x;

                 TarDat[number_ones].Y=y;

                 TarDat[number_ones].amp=max[index];
                 average[index]=100;
                 }

            else

                 {

                 average[index]=0;
                 max[index]=0;
                 }

            }

TarDat[number_ones+1].X=(-1);

TarDat[number_ones+1].Y=(-1);

return(number_ones);

}
```

REFERENCES

1. P.L. Chu, "Efficient CFAR Detection of Line Segments in a 2-D Image," *Proceedings ICASSP 1987*, Vol. 1, April 1987, pp. 587-590.

2. B. Porat and B.J. Friedlander, "A Frequency Domain Approach to Multiframe Detection and Estimation of Dim Targets," *Proceedings ICASSP 1987*, Vol. 1, April 1987, pp. 591-594.

3. Y. Barniv, "Dynamic Programming Solution for Detecting Dim Moving Targets," *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES 21, No. 1, January 1985, pp. 144-156.

4. N.C. Mohanty, "Computer Tracking of Moving Targets in Space," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. PAMI-3, No. 5, September 1981, pp. 606-611.

5. G.E. Bottomley, "The Effects of Cross-Correlated Noise and Multichannel Signal on ORing Loss," *Proceedings ICASSP 1987*, Vol. 2, April 1987, pp. 1103-1106.

6. W.A. Struzinski, "ORing Loss for Quantizers Followed by an ORing Device and an Accumulator," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-30, August 1982, pp. 668-671.

7. I.S. Reed, R.M. Gagliardi, and H.M. Shao, "Application of Three-Dimensional Filtering to Moving Target Detection," *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES-19, No. 6, November 1983, pp. 898-905.

8. S.D. Blostein and T.S. Huang, "Detection of Small Moving Objects in Image Sequences Using Multistage Hypothesis Testing," *Proceedings ICASSP 1988*, Vol. 2, April 1988, pp. 1068-1071.

9. D.L. Fried, "Statistics of Laser Beam Fade Induced by Pointing Jitter," *Appl. Opt.*, Vol. 12, No. 2, February, 1973, pp. 422-423.

10. H.L. Van Trees, *Detection, Estimation, and Modulation Theory*, New York, John Wiley and Sons, Inc., (1968).

11. M. Covell, (private communication, 22 July 1986).

12. W.H. Press, et al., *Numerical Recipes*, New York, Cambridge University Press (1986) p. 501.

13. "Loss in S/N Ratio in Peak Detecting N Channels of Data," ASW Technical Note, Hughes Aircraft Company, Fullerton, CA, 12 February 1968.

14. W.A. Struzinski, "Integration and Signal-to-Noise Ratio Requirements for a Signal Processing System Containing an OR-ing Device," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-31, June 1983, pp. 651-656.

15. W.A. Struzinski, "Performance Model for Square Law Detectors Followed by Accumulators and ORing Device," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-31, June 1983, pp. 759-761.

16. A.H. Nuttall, "Detection Performance Characteristics for a System with Quantizers, OR-ing, and Accumulator," *J. Acoust. Soc. Am.*, Vol. 73, May 1983, pp. 1631-1642.

17. P.L. Chu, "Optimum Projection for Multidimensional Signal Detection," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 36, No. 5, May 1988, pp. 775-786.

18. M.I. Skolnik, *Radar Handbook*, New York, McGraw-Hill (1970) pp. 15-24 to 15-27.

19. J.V. DiFranco and W.L. Rubin *Radar Detection*, Dedham, MA, Artech House, Inc., (1980) pp. 497-515.

20. L.J. Cimini and S.A. Kassam, "Data Quantization for Narrowband Signal Detection," *IEEE Trans. Aerosp. Electron. Syst.*, Vol. AES-19, No. 6, November 1983, pp. 848-858.

21. Y. Chen, "On the Design of Suboptimal Matched Filters for Three-Dimensional Moving Target Detection," MIT Lincoln Laboratory Technical Report TR-795, 20 November 1987, DTIC AD-A188384.

22. A.E. Filip, (private communication, October 28, 1988).

23. Y. Chen, "Cramer-Rao Bounds on the Accuracy of Location and Velocity Estimations using CCD Optical Sensors," MIT Lincoln Laboratory Technical Report TR-777, 2 November 1987, DTIC AD-A188481.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) Technical Report 846	5. MONITORING ORGANIZATION REPORT NUMBER(S) ESD-TR-89-104
6a. NAME OF PERFORMING ORGANIZATION Lincoln Laboratory, MIT 6b. OFFICE SYMBOL *(If applicable)*	7a. NAME OF MONITORING ORGANIZATION Electronic Systems Division
6c. ADDRESS *(City, State, and Zip Code)* P.O. Box 73 Lexington, MA 02173-0073	7b. ADDRESS *(City, State, and Zip Code)* Hanscom AFB, MA 01731
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Systems Command, USAF 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State, and Zip Code)* Andrews AFB Washington, DC 20334	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 292 / PROJECT NO. 12424F / TASK NO. / WORK UNIT ACCESSION NO.
11. TITLE *(Include Security Classification)* Efficient Detection of Small Moving Objects	
12. PERSONAL AUTHOR(S) Peter L. Chu	
13a. TYPE OF REPORT Technical Report 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT *(Year, Month, Day)* 1989 July 21 15. PAGE COUNT 80
16. SUPPLEMENTARY NOTATION None	

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUB-GROUP	detection; CCD imaging; velocity filtering; binary integration; maximum value projection; matched filtering; satellite detection; moving target detection

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A signal processing problem encountered with many sensor systems having a wide field-of-view is detection of small, unresolved objects moving in a straight line amid stationary clutter. The wide field-of-view combined with the need to accurately pinpoint object positions imply that these sensors must have hundreds of thousands of samples in their output. To process this amount of data in a timely fashion, computationally efficient algorithms are a necessity.

In this report, a computationally efficient set of algorithms is described for detecting satellites, meteorites, and other moving objects using data from an optical telescope charge-coupled device (CCD) focal plane in the MIT Lincoln Laboratory Demonstration Surveillance System (DSS). The trade-off of lowered detection sensitivity for lower computational cost in the algorithm is quantitatively discussed. Major techniques employed include: sample normalization by temporal mean and standard deviation to suppress clutter; maximum value projection to reduce the dimensionality of the data; a two-stage matched filter detector which first nominates and then confirms signal candidates; and two-dimensional binary velocity filter.

The techniques should have practical application to other wide field-of-view sensors where moving object detection is important.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Lt. Col. Hugh L. Southall, USAF	22b. TELEPHONE *(Include Area Code)* (617) 981-2330 22c. OFFICE SYMBOL ESD/TML